Decision Table Editor

ECSE 458 D1/D2: Capstone Project

Fall 2023 / Winter 2024

Group 50

Julien Lefebvre - 260985990 - julien.lefebvre2@mail.mcgill.ca

Yazan Saleh - 260892738 - yazan.saleh@mail.mcgill.ca

Lucca Di Lullo - 260984108 - lucca.dilullo@mail.mcgill.ca

Justin Randisi - 260987866 - justin.randisi@mail.mcgill.ca

WESTF Primary Members

Robert Sabourin - robsab@gmail.com

Ben Simo - ben@qualityfrog.com

Claudiu Stoianof - claudiu.stoianof@gmail.com

Abstract

Our Capstone project, the Decision Table Editor, was developed using Python and an agile development approach structured around three-week sprints. The motivation behind this project is the need for a specialized tool that simplifies the creation and manipulation of decision tables, enhancing decision-making processes in various applications.

The primary goal was to establish a robust foundation in the initial semester and deliver functional increments of the product at the end of each sprint. Starting with a preparatory sprint followed by eight development sprints, we successfully designed and implemented a comprehensive browser-based decision table editor. This tool now allows users to create, save, open, and manage decision tables with enhanced features such as support for multiple data types, complex logic reductions, and user-friendly interfaces.

Throughout the project, we extended the initial functionalities to include advanced features like logic optimization algorithms and the integration of environmental and societal impact analyses. The agile methodology facilitated a flexible, responsive development process, enabling us to adapt to technical challenges and feedback efficiently.

In conclusion, our project delivered a versatile and practical decision table editor that not only meets the initial requirements but also provides a solid platform for future enhancements. This final semester has culminated in a product that effectively supports complex decision-making, demonstrating the value and impact of our development efforts.

Table of Contents

0.0. List of abbreviations/notation used in the report

- General Project Terms:

i.	DT:	Decision Table
ii.	IDE:	Integrated Development Environment

- Technical Terms:

i.	JSON:	JavaScript Object Notation
ii.	CRUD:	Create, Read, Update, Delete

- Python and Web Development Terms:

i.	PyWebIO:	Python Web Input Output
ii.	UI:	User Interface

1.0. Introduction/Motivation/Objective

1.1. Introduction

In the realm of software development and data analysis, decision tables serve as a tool for defining and managing complex business logic and rules. Our Capstone project has successfully developed a specialized decision table editor designed to facilitate the creation, editing, and management of decision tables. Utilizing Python and an agile development approach, we have built a tool that enhances functionality and user experience, effectively addressing the needs identified at the project's start.

1.2. Motivation

The motivation for our project stemmed from a noticeable gap in efficient and user-friendly tools for handling decision tables. Many existing solutions lacked intuitiveness and failed to meet the diverse needs of users across various domains. Our decision table editor addresses these issues, offering enhanced ease of use, flexibility, and efficiency. It caters to both seasoned users of decision tables and newcomers, simplifying and streamlining workflows across different scenarios.

1.3. Objectives

The primary objectives of our project were to:

1. Develop a browser-based decision table editor that is intuitive and efficient for end-users.

2. Support a comprehensive range of functionalities including creating, saving, opening, and managing decision tables with various actions, conditions, and rules.

3. Create a scalable and adaptable tool that can evolve with user needs and technological advancements.

We have successfully achieved these objectives, delivering a tool that not only facilitates software development but also serves as an educational resource in academic settings.

1.4. Potential Applications

The decision table editor has broad applicability across multiple industries including software engineering, business process modeling, data analysis, and education. For example, in software engineering, it streamlines the translation of complex business rules into code. In educational contexts, it serves as a practical tool for teaching logical decision-making.

1.5. Future Scope

As we look to the future, our project is well-positioned for further enhancements. The agile nature of our development process, combined with ongoing feedback, allows us to continually refine and expand our tool. We anticipate that future teams will continue to incorporate more advanced features and adapt to new technological trends to meet emerging needs, building upon our foundational work.

2.0. Background

2.1. Overview of decision tables

Decision tables are a structured way of representing conditional logic, where complex decision rules are systematically laid out in a tabular format. These tables are composed of conditions, actions, and rules. Conditions are criteria or situations under which specific actions are taken. Actions are the responses or operations carried out when the conditions are met. Rules are the combinations of conditions and their corresponding actions [1].

2.2. Historical Context

Originating from the area of business decision-making and software engineering, decision tables have been a tool for encapsulating logical relations. Over time, their application has expanded into various fields, serving as a bridge between business logic and technical implementation.

2.3. Relevance in Software Development

In software development, decision tables play a crucial role in requirements engineering and business rules management. They offer a clear way to handle complex decision algorithms, ensuring that all scenarios are accounted for and managed efficiently. The management of these tables can become challenging with increasing complexity, particularly in aspects of scalability, clarity, and adaptability, all of which our project seeks to address [2].

2.4. Technical Foundations

Python was selected as a coding tool for its robustness and suitability for web-based applications, which aligns with our goal to create an accessible, browser-based DT editor. Furthermore, Python's extensive libraries and community support make it an ideal choice for developing a user-friendly interface with complex backend logic.

The agile methodology was chosen to manage the project due to its use of iterations. This methodology allowed for flexibility in development, enabling the team to adapt quickly to feedback and evolving project requirements [3].

2.5. Existing Tools and Comparisons

A review of existing DT editors highlighted a range of functionalities but also revealed gaps in user experience and versatility. Many existing tools are either too complex for novice users or too simplistic for advanced scenarios. Our project seeks to find a balance, offering a comprehensive yet intuitive interface for managing decision tables.

2.6. Summary

Understanding the theoretical aspects of decision tables, their evolution, and their application in software development is crucial. Mixed with the technical knowledge of Python and agile methodologies, this background forms the foundation upon which our project is built. It provides the necessary context to appreciate the motivations, challenges, and objectives of our DT editor project.

3.0. Requirements and Problem

3.1. Problem Description

The primary challenge our project addresses is the absence of an efficient, scalable, and user-friendly decision table editor tailored for complex logic management. Current solutions in the market either lack in functionality, suffer from complex user interfaces, or do not scale well with increasing complexity. This gap presents significant challenges in various fields that rely on decision tables for logical decision-making processes.

3.2. Project Requirements

3.2.1. Functional Requirements

Our decision table editor is designed with the following core functionalities:

1. *Creation and Editing:* Ability to create and modify decision tables, accommodating a variable number of conditions and actions (CRUD).

2. *Data Handling:* Support for various data types and structures within decision tables.

3. *Save and Export Options:* Facility to save work in-progress and export decision tables in widely-used formats like JSON.

4. *User Interaction:* A user-friendly interface that allows for easy manipulation and viewing of decision tables, including features like drag-and-drop and real-time updates.

3.2.2. Non-Functional Requirements

The project also adheres to several non-functional requirements:

1. *Usability:* The editor should be intuitive, catering to both novice and expert users.

2. *Performance:* Efficient performance even when handling complex decision tables with numerous conditions and actions.

3. *Security:* Ensuring the security and integrity of the data handled by the editor.

4. *Scalability:* Capability to manage an increase in user numbers or table complexity without performance degradation.

3.3. Constraints

3.3.1. Technical Constraints

Our project is constrained by the choice of Python for backend development and a web-based interface for frontend. The application must be compatible across major web browsers and ensure efficient data storage and retrieval mechanisms.

3.3.2. Resource Constraints

Given the limited timeframe of two semesters and a team of four, time management and distribution of tasks are crucial. Budget constraints also limit our choices regarding third-party services or tools.

3.3.3. Environmental Constraints

The project development process is subject to remote collaboration requirements, requiring effective online communication and coordination tools.

3.4. Summary

Understanding the outlined problem, the specific requirements, and constraints has been crucial for the development of our DT editor. These elements not only shaped the scope of our project but also guided our design and development strategies. By successfully meeting and in some cases, exceeding these requirements, our final product effectively addresses the identified needs and challenges, confirming the viability and necessity of our solution in practical settings.

4.0. Design and Results

4.1. Design Decisions and Process

4.1.1. Initial Design Approach

Our project began with a focus on Python for backend development and the PyWebIO library for creating a web-based UI, chosen for its compatibility with Python and its community support.

4.1.2. Decision-Making Process

We utilized Agile methodologies for our design process, focusing on iterative development and feedback incorporation through regular meetings with our team and advisors.

4.1.3. Key Design Decisions

1. *Library Selection:* We chose PyWebIO for its effective web application development capabilities.

2. *Feature Development:* In Sprint 1, we focused on essential functionalities for creating decision tables, actions, rules, and conditions so that a viable product could be launched from the start.

4.2. Results Obtained

4.2.1. Sprint 1 Progress

- Developed a functional prototype capable of creating decision tables.
- Implemented binary input (True/False) for conditions and actions, allowing real-time toggling by users.
- Introduced naming functionalities for decision table elements, enhancing user interaction and clarity.
- Enabled centralization of data in decision tables to facilitate saving and loading from personal storage.

4.2.2. Sprint 2 Progress

- Adjusted decision table structures and functionalities based on feedback from Sprint 1.
- Initiated development of an executable version of the application for easier access.
- Started implementing support for numerical ranges and values for conditions and actions, expanding the editor's capabilities.

4.2.3. Sprint 3 Progress

- Introduced custom types for user-defined data handling.
- Enabled file loading, allowing users to resume work on existing decision tables.
- Upgraded condition and action functionalities for enhanced table management.

4.2.4. Sprint 4 Progress

- Further refined the user interface for improved clarity and user workflow.
- Added the ability to modify conditions and actions within the tables.
- Implemented functionalities to delete rules and duplicate tables for testing scenarios.

4.2.5. Sprint 5 Progress

- Introduced the feature for renaming decision tables, conditions, and actions for better context.
- Added the functionality for users to enter condition values and action values within rules.
- Enhanced logic removal capabilities to streamline the decision logic and simplify the interface.

4.2.6. Sprint 6 Progress

- Optimized the application's performance for handling decision tables with numerous conditions and actions.
- Improved scalability features to ensure the editor remains efficient with increased user loads.

4.2.7. Sprint 7 Progress

- Focused on logic reduction techniques, adding the ability to combine rules and remove redundant ones.
- Started to incorporate E2GRULEWRITER algorithms to optimize decision tables further.
- Prepared and completed the project presentation.

4.2.8. Sprint 8 Progress

- Finalized the testing phase, with emphasis on usability and performance optimization.
- Prepared the project poster for the Design Day presentation, showcasing the project's capabilities and outcomes.

4.2.9. Final Working Product

Our application now provides a comprehensive suite for decision table creation and management. Users can easily navigate through a well-designed interface to manage complex decision logic.

Upon launching the application, the user is prompted with two buttons offering the choice to create a decision table or open an existing decision table stored locally as a JSON file as shown in Figure 1 below. After their selection, the user will see either a new and empty DT or a previously created DT.

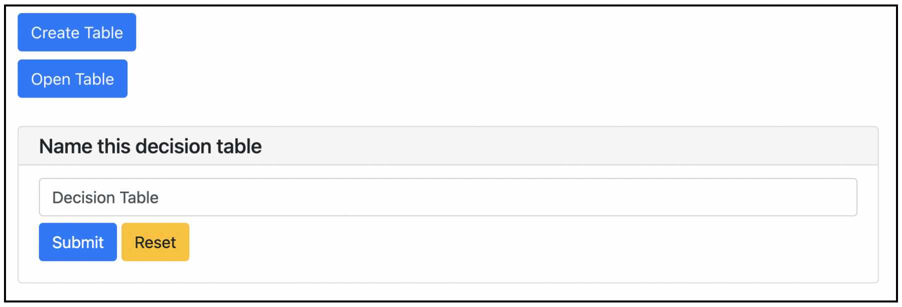
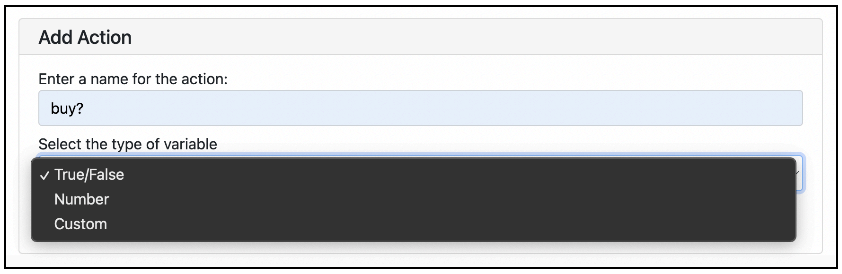


Figure 1: Screenshot of the application at the naming stage for a new decision table

Once working with an existing DT or a newly created one, the user can start to edit the content of the table. For example, the user can add new conditions, actions and rules, or edit those already present in the DT. Figure 2 below shows the prompt that can be seen by the user when creating a new action, in which the user can provide a name and a type for the action.


Add Action

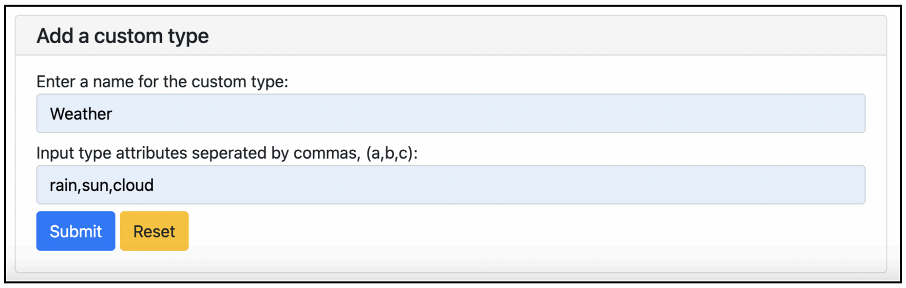
Enter a name for the action:

buy?

Select the type of variable

Figure 2: Screenshot of the application for an example of creating an action, choosing a name and a type

The editor provides the user with many different data types to use, including integers, decimal numbers, ranges and boolean values. If none of these types can suit a specific need for a condition or an action, the user can also create custom data types. Figure 3 shows the process of creating a custom type, which simply involves giving a name and a list of attributes that define it.


Add a custom type

Enter a name for the custom type:

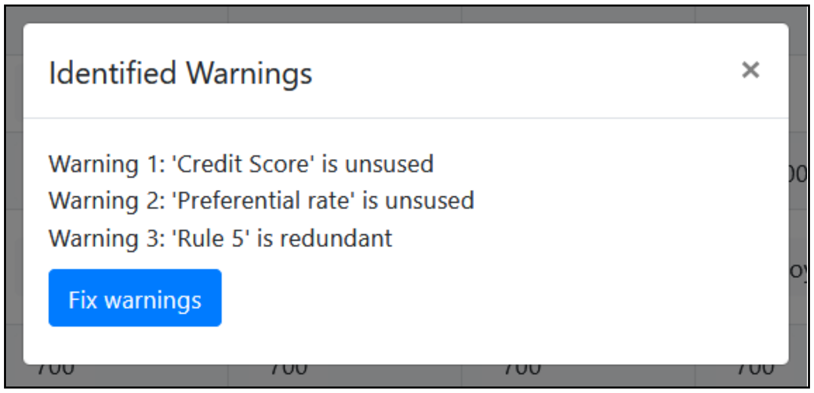
Weather

Input type attributes seperated by commas, (a,b,c):

rain,sun,cloud

Figure 3: Screenshot of the application at an example for creating a custom type

When using the table optimization features of the editor, the user may receive warnings about ineffective characteristics of the DT, as shown in Figure 4 below. In this example, the user is informed that the condition named ‘Credit Score’ does not affect any outcome of the table, that the action named ‘preferential rate’ is not used in the rules, and that rule 5 is redundant. The user can then decide to go back to the editor and fix those warnings manually, or use the *Fix Warnings* button to do so in a single click.


Identified Warnings

Warning 1: 'Credit Score' is unsused

Warning 2: 'Preferential rate' is unsused

Warning 3: 'Rule 5' is redundant

Figure 4: Screenshot of the application detecting a conflict in rules (logic optimization)

Our DT editor provides the user with nine different button options as shown below in Figure 5. These buttons cover all functionalities of the decision table and reflect the ease of use of our application for any type of user due to its straightforward approach.

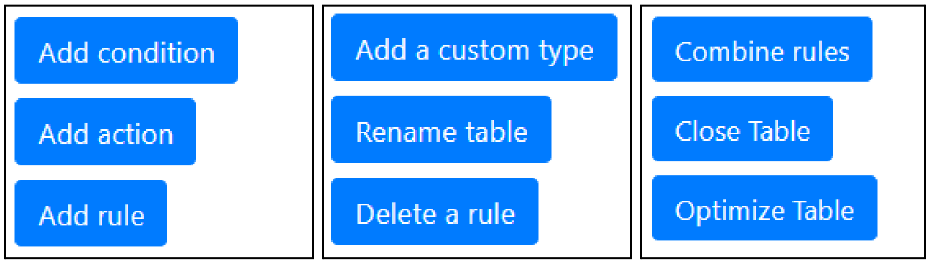
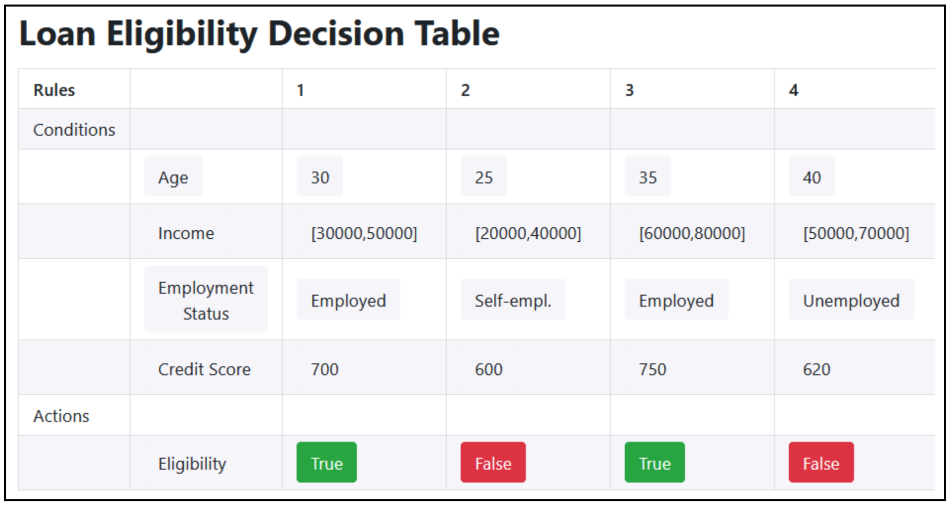


Figure 5: Screenshot of all the buttons available to the user in the DT application

An example of a fully functional decision table in practical use is shown below in Figure 6. In this example, we are using the decision table to determine whether an individual is eligible for a loan. Let us walk through an example to understand the helpfulness of a DT. If we have an individual whose age is 35, their income is 70000, they are employed and have a credit score of 750, then the action of eligibility is given as true. This implies the individual is in fact eligible and it was determined with the aid of a DT.



Loan Eligibility Decision Table

Rules		1	2	3	4
Conditions					
	Age	30	25	35	40
	Income	[30000,50000]	[20000,40000]	[60000,80000]	[50000,70000]
	Employment Status	Employed	Self-empl.	Employed	Unemployed
	Credit Score	700	600	750	620
Actions					
	Eligibility	True	False	True	False

Figure 6: Screenshot of the application showing an example of a complete decision table

4.3. Testing

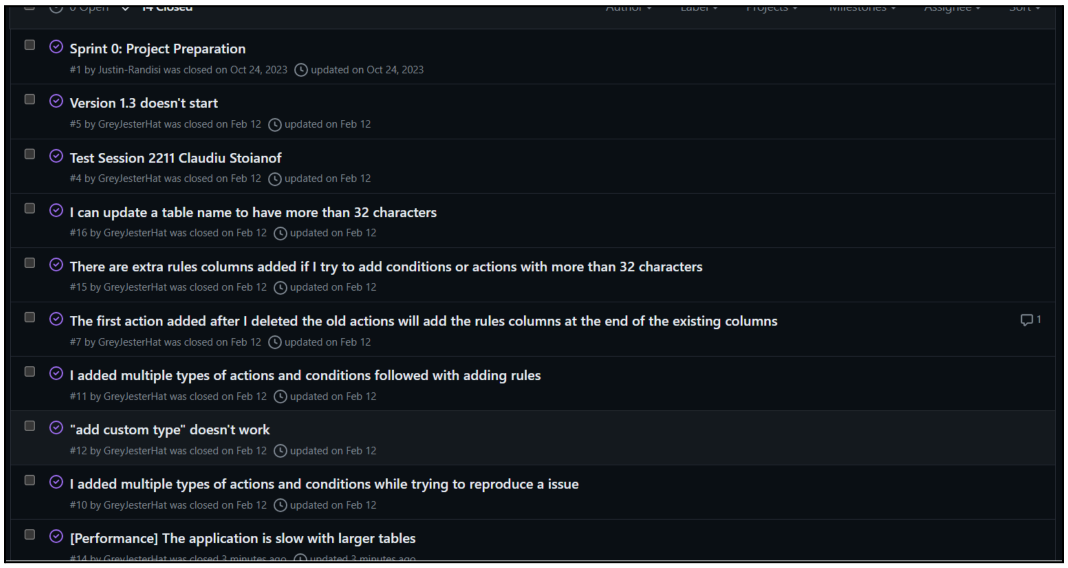
4.3.1. Testing plan

Our testing strategy was both dynamic and thorough. Following the completion of each sprint, WESTF team members provided targeted feedback highlighting potential areas for improvement, locations of bugs, and suggestions for new functionalities. Concurrently, Group 50 implemented a robust internal testing regimen, where members alternated in testing each other's contributions. This strategy ensured comprehensive coverage and maintained an unbiased evaluation of the application's functionality.

Transitioning from the initial plan to use Gherkin feature files, we adopted a hands-on testing approach that better suited our project's needs. This pivot was driven by the practical benefits and direct feedback mechanism provided by routine code reviews. These reviews, conducted by both the WESTF team and our internal members, served as a real-world gauge of the application's performance and usability. In conjunction with these reviews, the use of GitHub issues became an integral part of our testing framework, allowing us to meticulously document, track, and address any bugs or issues identified during testing. This approach not only facilitated immediate and actionable improvements but also fostered a collaborative environment for problem-solving and feature enhancement.

4.3.2. Testing results

The results of our comprehensive testing regimen were exceptionally positive. Each session of feedback and issue tracking led to direct enhancements of the application, increasing its robustness and ensuring a high level of user satisfaction. While no major bugs were encountered, the minor issues identified through GitHub were promptly resolved, with fixes implemented in the next development cycle. Figure 7 below shows some of the closed Github issues. This process of continuous testing and improvement played an indispensable role in achieving an application that adhered to stringent standards of reliability, performance, and user experience.



Sprint 0: Project Preparation
#1 by Justin-Randisi was closed on Oct 24, 2023 updated on Oct 24, 2023

Version 1.3 doesn't start
#5 by GreyJesterHat was closed on Feb 12 updated on Feb 12

Test Session 2211 Claudiu Stoianof
#4 by GreyJesterHat was closed on Feb 12 updated on Feb 12

I can update a table name to have more than 32 characters
#16 by GreyJesterHat was closed on Feb 12 updated on Feb 12

There are extra rules columns added if I try to add conditions or actions with more than 32 characters
#15 by GreyJesterHat was closed on Feb 12 updated on Feb 12

The first action added after I deleted the old actions will add the rules columns at the end of the existing columns 1
#7 by GreyJesterHat was closed on Feb 12 updated on Feb 12

I added multiple types of actions and conditions followed with adding rules
#11 by GreyJesterHat was closed on Feb 12 updated on Feb 12

"add custom type" doesn't work
#12 by GreyJesterHat was closed on Feb 12 updated on Feb 12

I added multiple types of actions and conditions while trying to reproduce a issue
#10 by GreyJesterHat was closed on Feb 12 updated on Feb 12

[Performance] The application is slow with larger tables

Figure 7: A screenshot of some of the closed GitHub issues tab

4.4. Tool Suitability and Integration

4.4.1. PyWebIO Integration

PyWebIO was instrumental in developing the interactive web interface of our decision table editor. Its simplicity and Python compatibility allowed us to focus on feature development rather than technical hurdles, proving invaluable for rapid prototyping and UI design.

4.4.2. Development Environment and Version Control

The Visual Studio Code (VS Code) IDE was the development environment of choice for all team members, providing a robust and versatile platform for coding. Its integrated support for Python and web development streamlined our workflow, while its extension ecosystem enhanced productivity.

GitHub served as the backbone for version control, issue tracking, and collaborative code integration. Its robust platform enabled us to manage our codebase effectively, track progress transparently, and facilitate peer reviews and merges with ease. The use of GitHub Issues was particularly effective for identifying, discussing, and resolving bugs and feature requests, forming a clear record of our project's evolution.

4.4.3. Documentation and Project Management

Google Drive played a pivotal role in documentation and project management. It offered a centralized repository for all project-related documents, allowing for real-time collaboration, version history, and access control. This facilitated an organized and accessible structure for storing sprint plans, progress reports, meeting minutes, and critical project artifacts.

4.4.4. Tool Assessment

The combined use of Python, PyWebIO, VS Code, GitHub, and Google Drive fostered a cohesive development environment that aligned well with our project goals and timelines. Each tool was chosen not only for its individual strengths but also for how effectively they integrated together, creating a streamlined development process that enhanced our productivity and collaboration. This synergistic toolset was crucial in developing a user-friendly, functional prototype within our project's constraints.

4.5. Summary

The project's design and results represent a cohesive progression through eight sprints, culminating in an advanced decision table editor. Strategic choices in tooling, adherence to agile practices, and an evolving testing strategy have resulted in a product that is both robust and user-centric. The final iteration stands as a versatile tool, ready to enhance decision-making processes across various domains.

5.0. Impact on Society and the Environment

Our decision table editor project offers a digital solution that influences societal operations and environmental conservation. This software tool transitions tasks typically done on paper onto a digital platform, aiming to enhance decision-making processes while being mindful of its ecological and social footprint.

5.1. Use of non-renewable resources

The creation of our decision table editor relies mainly on digital resources, reducing the consumption of non-renewable materials like paper and plastic. The energy used in development, largely from computers and servers, is the main resource expenditure. For users, the software requires minimal resources beyond the electricity to run it, sidestepping the environmental cost of manufacturing and waste disposal inherent in physical goods.

5.2. Environmental Benefits

Transitioning to our digital tool lessens the dependency on physical resources, lowering waste and conserving materials. Designed to be efficient, the software aims to use less computational power, which can lead to lower energy use than more demanding applications, thus having a smaller environmental impact.

5.3. Safety & Risk

The digital nature of our decision table editor inherently carries minimal physical risk. As the software operates locally without storing user data, the risk of data breaches is significantly mitigated. The primary safety consideration is ensuring the tool's reliability; inaccurate decision tables could lead to poor decision-making if the logic is not correctly applied. To this end, we emphasize the importance of rigorous testing and validation to ensure the integrity of the decision-making process facilitated by our software.

5.4. Benefits to Society

Our tool's societal benefits are primarily in its ability to make complex decision-making easier and more efficient in fields such as business, education, and software engineering. The enhanced decision-making has the potential to improve outcomes, increasing overall well-being. Economically, it saves time and resources, potentially driving growth in industries that utilize decision tables.

In summary, our decision table editor has a net positive impact on society by improving operational efficiency and potentially enriching quality of life. It also aligns with sustainable practices, reducing reliance on non-renewable resources. As the tool evolves, we hope any future teams assigned to our project are committed to assessing its impact on society and the environment, ensuring responsible contributions to technology and sustainability.

6.0. Report on Teamwork

Throughout the project, teamwork was pivotal to our success. Each member contributed significantly, ensuring a balanced and collaborative effort in developing the decision table editor. Our teamwork strategy involved weekly meetings and check-ins to assign tasks equitably, rotate challenging user stories, and support each other in completing sprint deliverables.

We utilized a branching strategy in our GitHub repository to facilitate individual contributions, merging our work in group sessions. The distribution of tasks is detailed in Table 1 below, reflecting a commitment to equal participation and diverse experience within the team. The complete backlog and description of stories can be found in appendix A.

Task	Person in charge	Sprint Completed
Backlog management	Yazan	All Sprints
Google Drive Documentation	Yazan	All Sprints
User Story 1	Julien	1
User Story 2	Julien	1
User Story 3	Lucca	1
User Story 4	Justin	1
User Story 5	Julien / Yazan	2
User Story 6	Yazan	2
User Story 7	Justin	2
User Story 8	Justin	2
User Story 9	Justin	3
User Story 10	Lucca	3
User Story 11	Lucca	3
User Story 12	Julien	3
User Story 16	Julien / Lucca	4
User Story 18	Yazan / Justin	4
User Story 13	Julien	5
User Story 14	Justin	5
User Story 15	Yazan	5
User Story 20	Lucca	5
User Story 21	Lucca	5
User Story 22	Julien	5
User Story 23	Justin	5
User Story 24	Yazan	5
User Story 28	Lucca	6

User Story 29	Julien	6
User Story 25	Julien	7
User Story 27	Lucca / Yazan	7
User Story 31	Justin / Yazan	7
User Story 26	Yazan	8
User Story 30	Lucca / Julien	8
User Story 32	Justin	8
User Story 33	Justin	8

Table 1: Work distribution of all the tasks and user stories completed among the team members

Figure 8 below presents a bar chart summarizing each member's time contribution to user stories. On average, each member dedicated approximately 116 hours to user stories over 22 valid weeks (excludes holiday weeks), equating to about 5.3 hours per week per member on user stories alone. This figure does not include time spent on research, documentation, testing, and meetings, which, when factored in, brings the average total weekly contribution to over 8 hours per week per member.

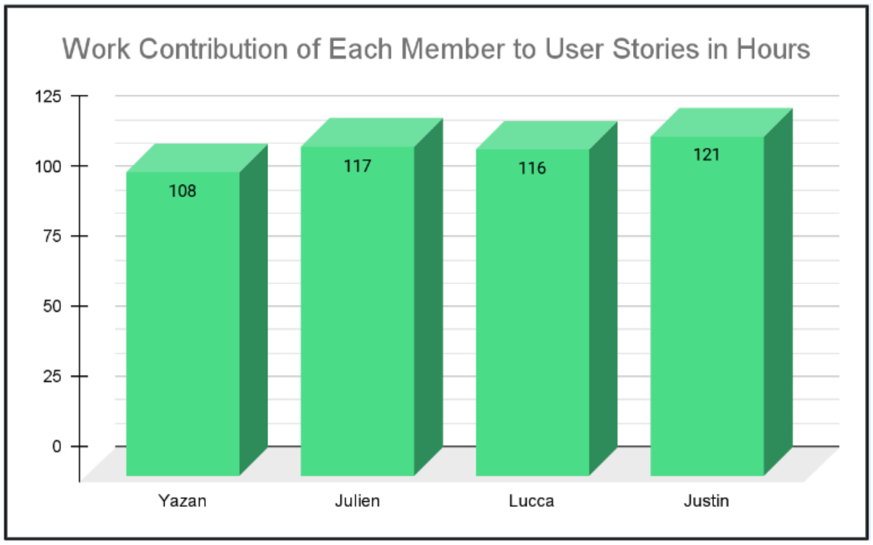


Figure 8: Bar Chart displaying the corresponding contributions of each member to user stories in hours

This demonstrates a significant and consistent investment of time and effort from each team member in all project aspects. Our commitment extends beyond the development tasks to include meticulous documentation in Google Drive and comprehensive testing and problem-solving in group settings. The agile communication within the team has been a cornerstone of our process, ensuring prompt and effective exchanges.

As we move forward, we intend to maintain these productive team dynamics, which have been crucial to delivering a product that not only meets our collective standards but also one that we take pride in. Our methodical and cooperative approach will continue to be a defining factor in our project's ongoing success.

7.0. Conclusion

The journey of our decision table editor project has reached a significant milestone. Over the course of two semesters, we have transformed a concept into a functional and intuitive tool designed to enhance complex decision-making. Our accomplishments extend beyond the initial development of a prototype; we have created a feature-rich application that allows users to interact with, manage, and optimize decision tables effectively.

The strategic choice to use Python and the PyWebIO library has proven effective, providing the flexibility and power needed to build a robust tool. Our iterative development process, anchored by agile principles and consistent feedback, has allowed us to refine the editor to meet the nuanced needs of our users.

As we conclude this phase, we look forward to the project's future. We are preparing to hand over a comprehensive suite of documentation and the complete codebase to our supervisor, Robert Sabourin. This will enable a seamless transition to a new team who will continue to refine and expand upon our work, ensuring the decision table editor remains a valuable asset within the software development toolkit.

Throughout this project, our team has not only advanced technically but has also matured in our collaborative and problem-solving abilities. The experience has reinforced the vital role of user-centric design and the balance between technical innovation and practical usability.

In summary, our project has not only achieved its initial goals but has also set a precedent for future development. It stands as a testament to the power of collaborative effort and adaptive planning in software engineering. As we pass the torch to the next team, we are confident that the decision table editor will continue to evolve and positively impact its users' work.

8.0. References

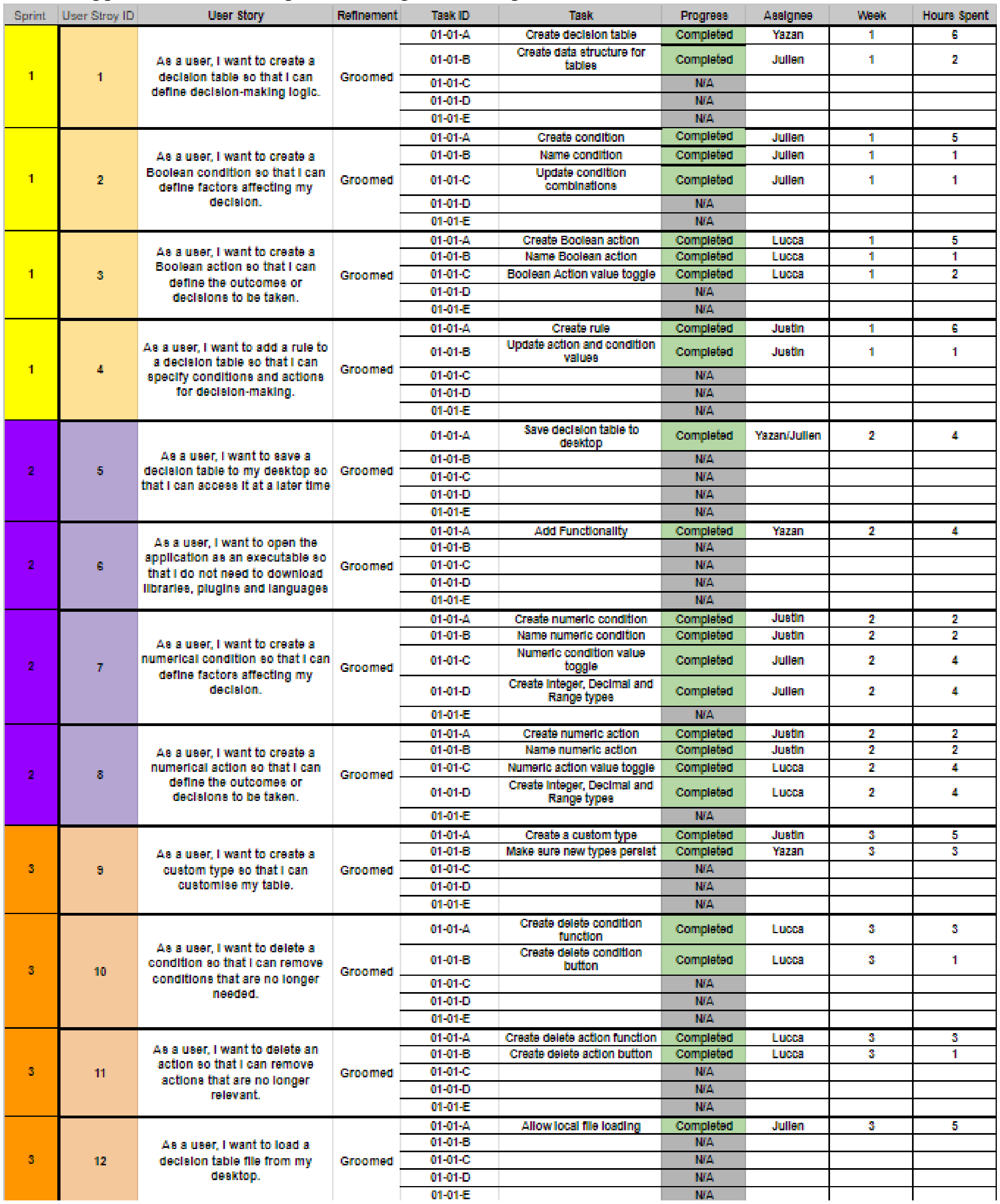
[1] *Decision tables*. IBM.(2024, Feb 16) https://www.ibm.com/docs/en/odm/8.12.0?topic=tables-decision, [Last accessed: 2024, Apr 11]

[2] K. Wiegers and J. Beatty, (2013). *"Software Requirements",* 3rd ed., Microsoft Press,

[3] I. Sommerville, (2015). *"Software Engineering,"* 10th ed., Pearson Education,

9.0. Appendices

Appendix A: The complete backlog and description of stories



Sprint	User Stroy ID	User Story	Refinement	Task ID	Task	Progress	Assignee	Week	Hours Spent
1	1	As a user, I want to create a decision table so that I can define decision-making logic.	Groomed	01-01-A	Create decision table	Completed	Yazan	1	6
				01-01-B	Create data structure for tables	Completed	Julien	1	2
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
1	2	As a user, I want to create a Boolean condition so that I can define factors affecting my decision.	Groomed	01-01-A	Create condition	Completed	Julien	1	5
				01-01-B	Name condition	Completed	Julien	1	1
				01-01-C	Update condition combinations	Completed	Julien	1	1
				01-01-D		N/A			
				01-01-E		N/A			
1	3	As a user, I want to create a Boolean action so that I can define the outcomes or decisions to be taken.	Groomed	01-01-A	Create Boolean action	Completed	Lucca	1	5
				01-01-B	Name Boolean action	Completed	Lucca	1	1
				01-01-C	Boolean Action value toggle	Completed	Lucca	1	2
				01-01-D		N/A			
				01-01-E		N/A			
1	4	As a user, I want to add a rule to a decision table so that I can specify conditions and actions for decision-making.	Groomed	01-01-A	Create rule	Completed	Justin	1	6
				01-01-B	Update action and condition values	Completed	Justin	1	1
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
2	5	As a user, I want to save a decision table to my desktop so that I can access it at a later time	Groomed	01-01-A	Save decision table to desktop	Completed	Yazan/Julien	2	4
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
2	6	As a user, I want to open the application as an executable so that I do not need to download libraries, plugins and languages	Groomed	01-01-A	Add Functionality	Completed	Yazan	2	4
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
2	7	As a user, I want to create a numerical condition so that I can define factors affecting my decision.	Groomed	01-01-A	Create numeric condition	Completed	Justin	2	2
				01-01-B	Name numeric condition	Completed	Justin	2	2
				01-01-C	Numeric condition value toggle	Completed	Julien	2	4
				01-01-D	Create Integer, Decimal and Range types	Completed	Julien	2	4
				01-01-E		N/A			
2	8	As a user, I want to create a numerical action so that I can define the outcomes or decisions to be taken.	Groomed	01-01-A	Create numeric action	Completed	Justin	2	2
				01-01-B	Name numeric action	Completed	Justin	2	2
				01-01-C	Numeric action value toggle	Completed	Lucca	2	4
				01-01-D	Create Integer, Decimal and Range types	Completed	Lucca	2	4
				01-01-E		N/A			
3	9	As a user, I want to create a custom type so that I can customise my table.	Groomed	01-01-A	Create a custom type	Completed	Justin	3	5
				01-01-B	Make sure new types persist	Completed	Yazan	3	3
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
3	10	As a user, I want to delete a condition so that I can remove conditions that are no longer needed.	Groomed	01-01-A	Create delete condition function	Completed	Lucca	3	3
				01-01-B	Create delete condition button	Completed	Lucca	3	1
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
3	11	As a user, I want to delete an action so that I can remove actions that are no longer relevant.	Groomed	01-01-A	Create delete action function	Completed	Lucca	3	3
				01-01-B	Create delete action button	Completed	Lucca	3	1
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
3	12	As a user, I want to load a decision table file from my desktop.	Groomed	01-01-A	Allow local file loading	Completed	Julien	3	5
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			

4	16	As a user, I want to modify a condition so that I can make changes to existing condition definitions.	Groomed	01-01-A	Add button	Completed	Julien	4	3
				01-01-B	Add functionality	Completed	Lucca	4	5
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
4	18	As a user, I want to modify an action so that I can make changes to existing action definitions.	Groomed	01-01-A	Add button	Completed	Yazan	4	3
				01-01-B	Add functionality	Completed	Justin	4	5
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	13	As a user, I want to close a decision table so that I can create a new one or end the session.	Groomed	01-01-A	Add functionality	Completed	Julien	5	3
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	14	As a user, I want to delete a rule from a decision table so that I can remove unnecessary rules.	Groomed	01-01-A	Add functionality	Completed	Justin	5	4
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	15	As a user, I want to duplicate a decision table so that I can create a copy for experimentation or variation.	Groomed	01-01-A	Add functionality	Completed	Yazan	5	4
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	20	As a user, I want to rename a decision table so that I can give it a more meaningful name.	Groomed	01-01-A	Add button	Completed	Lucca	5	4
				01-01-B	Add functionality	Completed	Lucca	5	2
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	21	As a user, I want to rename a condition so that I can provide a more descriptive name.	Groomed	01-01-A	Add button	Completed	Justin	5	2
				01-01-B	Add functionality	Completed	Lucca	5	2
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	22	As a user, I want to rename an action so that I can provide a more descriptive name.	Groomed	01-01-A	Add button	Completed	Julien	5	2
				01-01-B	Add functionality	Completed	Julien	5	2
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	23	As a user, I want to enter condition values in a rule so that I can define specific conditions for a rule.	Groomed	01-01-A	Add functionality	Completed	Justin	5	2
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
5	24	As a user, I want to enter action values in a rule so that I can specify the outcomes of a rule.	Groomed	01-01-A	Add functionality	Completed	Yazan	5	3
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
6	28	As a user, I want to remove unused conditions from a decision table to simplify the logic.	Groomed	01-01-A	Add Functionality	Completed	Lucca	6	3
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
6	29	As a user, I want to remove unused actions from a decision table to streamline decision logic.	Groomed	01-01-A	Add Functionality	Completed	Julien	6	3
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			

7	27	As a user, I want to combine rules in a decision table so that I can simplify the decision logic.	Groomed	01-01-A	Add Functionality	Completed	Lucca	7	4
				01-01-B	Add Button	Completed	Yazan	7	1
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
7	31	As a user, I want to remove incomplete rules from a decision table to ensure accurate decision-making.	Groomed	01-01-A	Add incomplete rule detection	Completed	Justin	7	4
				01-01-B	Add fix of all incomplete rules	Completed	Yazan	7	2
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
7	25	As a user, I want to remove redundant rules in a decision table to streamline decision logic.	Groomed	01-01-A	Add incomplete rule detection	Completed	Julien	7	3
				01-01-B	Add fix of all incomplete rules	Completed	Julien	7	2
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
8	26	As a user, I want to identify conflicting rules in a decision table to resolve inconsistencies.	Groomed	01-01-A	Add Functionality	Completed	Yazan	8	3
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
8	30	As a user, I want to utilize algorithms from E2GRULEWRITER to optimize decision tables.	Groomed	01-01-A	Add Functionality	Completed	Lucca	8	4
				01-01-B	Add Functionality	Completed	Julien	8	4
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
8	32	As a user, I want to eliminate irrelevant conditions from a decision table to simplify logic.	Groomed	01-01-A	Add Functionality	Completed	Justin	8	2
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			
8	33	As a user, I want to eliminate redundant rules from a decision table to optimize decision logic.	Groomed	01-01-A	Add Functionality	Completed	Justin	8	2
				01-01-B		N/A			
				01-01-C		N/A			
				01-01-D		N/A			
				01-01-E		N/A			