U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Risk Matrix Creation Interface

Capstone Project
PE05

Authors
André Lino dos Santos (up201907879@edu.fe.up.pt)
Edgar Ferreira da Torre (up201906573@edu.fe.up.pt)
João Afonso Andrade (up201905589@edu.fe.up.pt)
Maria José Carneiro (up201907726@edu.fe.up.pt)

Supervisors
André Restivo (arestivo@fe.up.pt)
Henrique Cardoso (hlc@fe.up.pt)

July 2022

Contents

1 Introduction

This project was made in the context of the Capstone Project curricular unit. The selected theme was a proposal to develop an interface to create risk matrices and visualize the mapping of the different database variables in the used formulas. The work method consisted in a weekly iteration schedule, which was articulated in the meetings at the end of the week with the report on the work progress and followed by further planning.

These specifications were required to the LIACC - Laboratory of Artificial Intelligence and Computer Science at the University of Porto by ERS - Entidade Reguladora de Saúde, the entity responsible for the regulation of all the health care services in Portugal. This application would be useful on a daily basis, since a different approach could be taken in one of the areas of action of the ERS: looking at the complaints from the establishments. The complaints come in great amount, so it would be of great help the existence of an interface where all the information could be treated automatically and compiled in a way where patterns and structural problems are noticed with ease.

Later on, the project was integrated with a similar requirement of another entity: ASAE - Autoridade de Segurança Alimentar e Económica.

A prototype developed by LIACC had already been developed, but we were asked to provide a different approach on the project, so our work did not have any background and was started from zero. The initial project contained many different objectives, ranging from the database creation/modification to the matrix interface, so the focus was redirected to the generation of new formulas and their respective graph visualization. The carried work was developed in different phases:

- System Details Definition
- Interface Design
- Formula Menu Implementation
- Graph Visualization
- Mock Database Setup

Summing all up, the final product contains an interface capable of defining the formulas to use in the matrices in a simple way, accessible to users with few knowledge on the matter and from all areas of expertise, but still offering the option to create complex formulas to a fair extent. Every feature and detail will be thoroughly explained in the following sections.

The source code can be found here.

2 Design and Structure

2.1 Interface

The mock-ups defined initially contained perspectives for the majority of the different menus. Initially, the main screen of the platform would contain the connections to its features, those being the matrix visualisation section, the graph section, the saved data section and a small profile page to enable the ability to edit the user preferences.

After exchanging some ideas in the weekly meeting, an establishment of priorities was agreed by every part, so some of the initial ideas were left in stand-by (the profile page, for example). The graph and matrix menus also changed their role in the architecture, since the first was integrated inside the latter as a complement to the variable creation section, to achieve a more accessible flow (the user wants to see the graph impact when defining a risk matrix).

Figure 1: First Interface Draft

As the feature previously mentioned was the core to defining the whole formula definition interface, the project focus was directed into it. In later iterations, many other small changes were incrementally applied to that specific menu, which will be further explained in the implementation section.

Its design is composed by an input box to name the new variable, a visor that shows the current defined formula, the graph preview from the database and the edition section itself. The latter is divided in 6 main components per variable:

- Select variable from the database
- Select the function to apply in the variable
- Select the variable weight
- Select the example value to apply in the formula result in real-time
- Select if the variable should be Normalized in the Preview Graphs
- Delete variable

Figure 2: Formula Definition Control Interface

Below each of these, 2 buttons allow the user to add another variable, either by multiplying it or summing it to the previous one. As these buttons are interacted with, the formula gets more complex at the user's will, and the new variable gets assigned its values according to that. The graph is updated considering the formula and the respective variables values stored in the database.

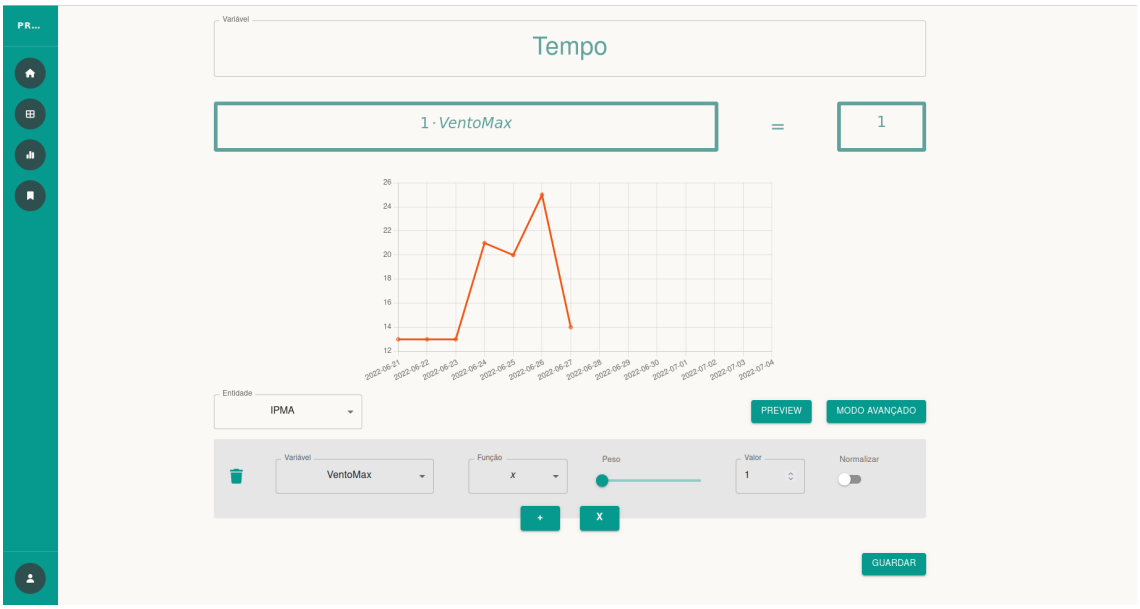

Figure 3: Final Interface

The user can also opt to examine the graph preview using general generated values. In this section, it is given the possibility to lock one of the variables to the graph's 'X' axis, providing sliders to the rest, which can be interacted with in real time to examine its impact in the 'Y' value, the new variable result.

As previously mentioned, the risk matrix creation menu was not implemented but the wire-frame was designed. In the plan there are different parts contemplated. Similarly to the formula menu, there are dropboxes that allow the choice of any variable in the database, including the manually created ones(the link to the creation page is located here). Complementing that, these variables also allow the definition of its weight and desired interval of values in the matrix. The time factor is also an available inclusion, giving extra insight on the risk matrix value evolution over time. Finally, the matrix has different edition preferences available, allowing the color scheme and the grid's number of squares to come as the user wishes.

Figure 4: Risk Matrix Creation Draft

2.2 Database

As the organization's data was confidential, its use was restricted in this project context, so a simple mock database was created to provide data to the graphs and simulate the function variation. Initially only get functions returning lists were being used, but later on an actual MySql database was integrated in the code, in order to get a more realistic approach. The structure was simply implemented as follows:

- Variables Table: Holds the variables that the system is tracking. Each row has an id (`INT`) which is the primary key, a variable name (`VARCHAR(200), NN`), the value (`DOUBLE`, defaulted to 0.0), the date the value was taken (`DATETIME`, defaulted to the current date) and the entity it corresponds to (`VARCHAR(200), NN`).
- Functions Table: Holds the functions to be applied to each variable. Each row has an id (`INT`) which is the primary key and the function to apply (`VARCHAR(45)`).

2.3 Technologies Used

In our first approach to this project we decided that we would primarily focus on using Django in order to manage our database use, defined in MongoDB. However, since the

scale of our project was small and we were only dealing with data mocks and the dimension of new technologies to learn was growing, we decided to implement our backend in NodeJS, mainly ExpressJS and manage our database mock with MySQL, which was a technology we were more familiar with.

As for our frontend, we decided to use React since it is wildly used in many different web applications and would allow us to develop a simple yet interactive and appealing interface for the final users. It also was a technology we had never used, so learning it was a challenging but useful experience.

3 Implementation

3.1 Formula Creation

The formula creation section is composed by smaller interactive units. Firstly focusing on the general details, the formula variable components, are stored in an array containing the list of the currently existing variable objects. Each entry contains all the information the object needs to show or calculate results in this menu(further explained below). Every time an alteration occurs, the array updates itself and applies it on the screen via state changes. The same happens to the operator storing array, which saves the operators used to connect all the variables. If the variable array has solely 1 item, the operator array will be empty. In case the first has 2 entities, the latter will have 1, and so on.

On another note ,the visor who shows the whole formula processing and conversion to its visualization form is handled by the 'react-equation' module. This allows the application to convert text expressions into improved and visually attractive formats, such as the fraction or square operators.

Figure 5: Formula Visualisation

Redirecting the attention to the variable edition options, there are 6 different possible interactions.

- The first one is the database variable selection dropdown. In the list, all the database variables are shown, either the original or the user created ones. The user may select which one of those is pretended to be applied.

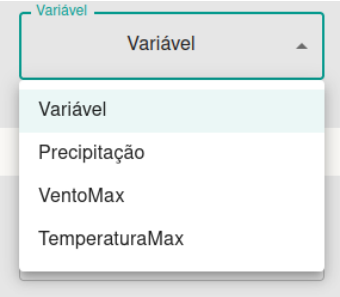

Figure 6: Variable Selection Dropdown

- In the second item, a similar architecture is taken since a drop down menu is also used. However, this time, the selection options are relative to the function to be applied in the variable. This menu is hard coded since the calculation module needs specific instructions to work, so only specifically inputted functions are available to use. Nevertheless, new options may be implemented accordingly to the client needs.

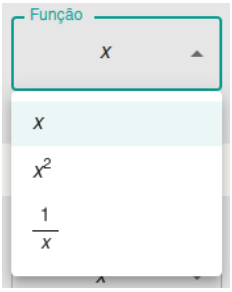

Figure 7: Function Selection Dropdown

- The next available edition option is the variable weight. It's fair to say that one may want to increase/decrease a variable's impact on the final result. Because of that, a slider is available so that this feature can be easily edited.

Figure 8: Weight Slider

- After the initial three, we start to get into more specific options. The fourth one does not impact the formula directly, but rather the simulated result on the screen. Just aside the formula container, there is the container responsible by simulating the result according to the the fourth parameter of each variable. It replaces the values defined on each input box by their respective variables on the formula. This way, we can interact with the application to understand how a greater value on a certain variable impacts the overall result.

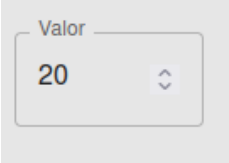

Figure 9: Value Selector

- The next one is also only applicable on another scope. When using the graph preview options there may be a need to normalize certain variables, if not all. Therefore, each of the variables has a switch that enables its normalization in the visualization mode, altering the graph.

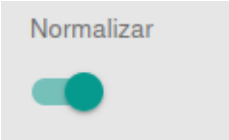

Figure 10: Normalization Switch

- Finally, the most simple of all, the delete button. It simply removes the current variable and its attached dependencies, applying the next operation to its ancestor as if the deleted variable never existed.

Figure 11: Delete Button

The plus and times operators are the only ones available to connect the variables in the formula, since the other 2 main operations are achievable with the function options. Typing on the last row of these buttons will create a new variable section with the default values, ready to be edited at free will, using the respective operator to connect the new variable to the last formula entity. Clicking on the already existent variable operators will change the one pressed, instead of creating a new one.

Figure 12: Operation Buttons

3.2 Graphs

Initially what was set out to be implemented graph-wise, was the database dependent variation of the formula value, in function of one of the variables(X-axis). However, after some idea exchange in one of the weekly meetings, the graph complement was split into two parts: one who kept the database values and another who used general values. The graph visualisation was implemented using `chart.js`.

3.2.1 Standard Preview Visualisation

The standard preview visualization is an option available just on top of the variable list. In this section, a general graph related with the X-axis locked and selected variable, showing the correspondent final result. All the other variable values are editable with a slider, and their change affects the graph in real-time. The selected variable is changeable. If the normalization switches were previously activated the sliders will be updated accordingly. The X-axis limits are borrowed from their respective value in the database. Overall this feature brings up a view on the function's evolution real time. If the selected variable is assigned to a square function, the showed curve will be a parabola, for example.

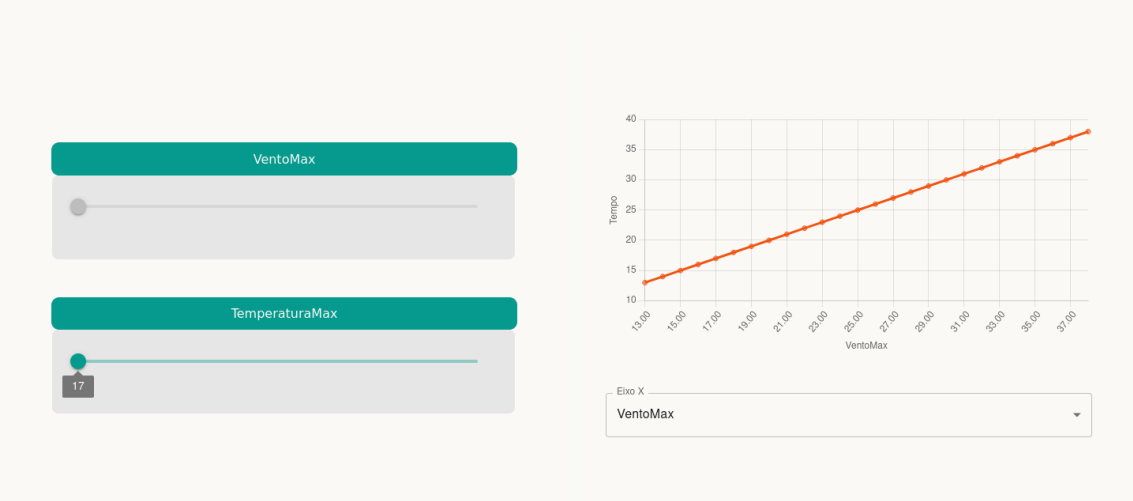

Figure 13: Standard Preview Visualization

3.2.2 Database Dependent Visualisation

This graph does not need to be opened in any pop-up, since its location is on the main formula creation page. The difference between the 2 lies on the variable values. Instead

of having a variable locked to the X-axis, the measure used are the data collection time units, so that a perception of the new variable evolution is brought up, in the previous two weeks. All the variable values are fixed, since they are read from the database. If there is no value defined for any of the last two weeks, no point is drawn on the chart. The formula evaluation is processed using the requested information and outputs the final value for each day (for example), building up the points that sustain the graph. It is also changed in real-time, in order to show the effect a variable change has on the output. This development was the last to be developed, since it was dependent on the mock-up database, which was also projected in the last iteration.

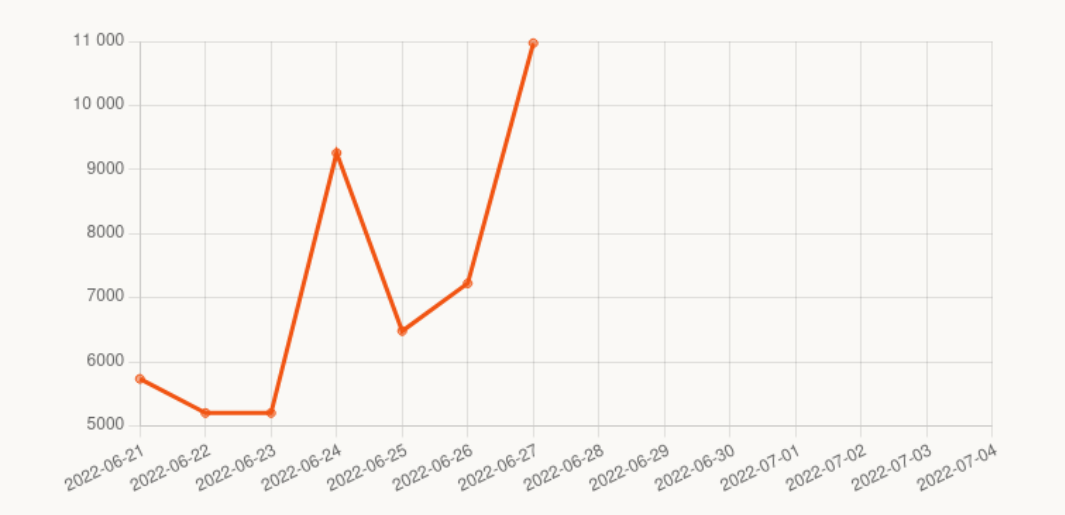

Figure 14: Database Dependent Visualization

4 Conclusion

With this project, we were able to develop a simplified and interactive visualization interface capable of defining complex variables by applying formulas to simpler ones and observing the impact of their own changes and interactions with other variables, through charts and calculated values. We explored a different approach to what had already been developed by the LIACC department in the context of their risk calculations prototype for the ERS, by creating a more general template useful to users from all areas of expertise with minimal knowledge of complex mathematical formulas, but with a necessity to analyse data and it's mutations. This project made it possible to visualize that new perspective and hopefully was a relevant contribution to improving the already developed software with a more approachable and user-friendly interface regarding variable constructions.

During the course of this project, we were able to learn more about risk calculations and it's applications in real problems and institutions. We are able to improve our Web Development skills as well as gaining experience in different technologies we had never used before and that are widely adopted by many big companies today. Furthermore, we strengthened our software designing skills, since planning and debating different approaches and solutions was a big part of our development process. We also improved our soft skills like team working, communication, organization and time management.

We think that this project gave us a set of skills that will certainly be useful in future projects and work environments. We also feel that the project could be expanded since some of the ideas we discussed initially were not implement due to time constraints, but the core of the new approach serves as a great comparison against what had already been developed, and will surely be a great asset. All in all, we are proud with what we achieved in this project and will surely benefit from it as future engineers.

5 Appendix

5.1 Iteration Log

5.1.1 End of March - Middle of April

- Study of risk matrices and it's IA applications in public administration
- Discussion about the work already developed by FEUP about the ERS solution of matrix visualization
- Defining the project's goal and key components: an interactive and simplified tool for designing and visualizing complex variables in order to evaluate their risk
- Mock ups and initial ideas
- Project's "Boilerplate" using the selected technologies

5.1.2 Middle of April - Middle of June

- Incremental implementation of the components regarding the discussed goal: general layout, formula creation and it's interactions.
- Development of the standard preview chart and it's integration with a previously created variable.
- Development of the database dependent chart and it's real time interaction with the formula created.
- General database mockup in a .js file.

5.1.3 Middle of June - Start of July

- General database mockup in a MySQL environment.
- Integration of the database with the developed visualization solution.
- Generalization of queries through a `JSON` format file, in order to integrate with the real confidential database.
- Small and final style changes.
- Documentation and delivery.

6 References

React Documentation, Available at: `www.reactjs.org`

Material UI Documentation, Available at: `www.mui.com`

Chart.js Documentation, Available at: `www.chartjs.org`

MySQL Documentation, Available at: `www.mysql.com`

Express.js Documentation, Available at: `www.expressjs.com`