DM

Extreme Weather Conditions Dashboard

MASTER DISSERTATION

Luís Carlos Gonçalves Freitas
MASTER IN INFORMATICS ENGINEERING

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

February | 2022

Extreme Weather Conditions Dashboard

MASTER DISSERTATION

Luís Carlos Gonçalves Freitas
MASTER IN INFORMATICS ENGINEERING

ORIENTATION
Filipe Magno de Gouveia Quintal

CO-ORIENTATION
Sabrina Scuri

Contents

Dedication

This thesis work is dedicated to my girlfriend, Diana, who has been a constant source of support and encouragement during the challenges of graduate school and life. I am truly thankful for having you in my life. This work is also dedicated to my parents and my sister, João, Guilhermina and Cristina respectively, who have always loved me unconditionally and whose good examples have taught me to work hard for the things that I aspire to achieve. Finally, I would like to thank my advisor Filipe Quintal and co-advisor Sabrina Scuri, for all the effort and dedication, as without them it would not have been possible.

Abstract

Our planet Earth is changing due to the action of Man. Several factors influence the functioning of our planet, such as the constant need for human evolution, population growth and many others. Due to this, more and more we have extreme atmospheric events.

There is an increasing need to become aware of this problem and take action both in terms of what we can do to prevent the destruction of our planet and prevent when these extreme events happen. The objective of this thesis is based on this last point.

To combat the lack of a system in Portugal capable of helping the general public in extreme weather conditions, we created an atmospheric dashboard. The goals were that this dashboard was simple to use, open to the general public, with high spacial resolution (currently the smallest area used is the municipality, here we use an even smaller geographic area), flexible, and scalable so that new data sources or new variables could be added to the existing ones in the future without affecting the correct functioning of the system.

This system uses meteorological data and relates them to other variables such as the age of the population, the thermal conditions of buildings, number of deaths, among others. The combination of all these data leads to the calculation of a risk index associated with each zone.

The results show that the application has excellent portability, as it was easily installed on another device thanks to the virtualization offered by Docker.

In terms of scalability, although did no tests at this level. However, during the application development, there was the need to add new data sources. Considering that the application is modularised, there was no problem with the existing or new data sources.

Users see great potential in the application in terms of usability for prevention by the population during extreme weather events. Another positive aspect was that they liked the layout of the elements in the interface, namely the fact that the map area is the most prominent and most visible area in the dashboard. It also helped us get some feedback on what could be improved, such as using more distinct colors to present the level of risk or even improving the connection between the different views of the application.

The results are positive and encourage further development of the application, applying the suggestions obtained from user testing.

Glossary

ADENE	Portuguese Agency for Energy. 3, 7, 25
API	Application Programming Interface. 34, 42, 47, 53–55, 59–61, 64, 67, 75, 86, 91, 92
CO2	Carbon Dioxide. 1, 4
CSS	Cascading Style Sheets. 26, 27
DGS	Direção-Geral de Saúde. 17
DMSS	Daily Mortality Surveillance System. 25
EPC	Energy Performance Certificates. 7
EWE	Extreme Weather Events. 7
FRIESA	Health Risk warning systems for cold and influenza. 5, 6, 25, 38, 83, 86
HTML	Hypertext Markup Language. 26, 27, 47
ICARO	Health Risk warning systems for heat. 5, 6, 24, 38, 83, 86
INSARJ	Instituto Nacional de Saúde Dr. Ricardo Jorge. 5, 25
IPMA	Instituto Português do Mar e da Atmosfera. 15, 24, 25, 87
MPA	Multi-Page Applciation. 29
NIS	National Institute of Statistics. 22
NSI	National Statistics Institute. 7
NTUSP	Nomenclature of Territorial Units for Statistical Purposes. 22
SCE	Certification of Buildings System. 6
SPA	Simple Page Applciation. 29
SQL	Structured Query Language. 33
UBD	Urban Building Database. 11, 12
UI	User Interface. 29

1 Introduction

The planet Earth is not the same as 30 or 40 years ago. The same goes for our future planet. This constant evolution has multiple implications. As the planet evolves, so does the human race and everything around us. The massive growth of the population has led to the over-exploitation of natural resources, bringing about issues like poverty, hunger and many others. Natural resources are limited, and at this rate of growth, they will soon no longer be enough to cope with the needs of our society. We are consuming the available resources faster than they can be replaced, thus, it is imperative to find new ways of sustainability for humankind [21] [35] [22].

Population growth requires continuous expansion of existing buildings and the construction of new ones. However, free plots of land are needed to sustain these urbanization trends. Humans are therefore destroying forest areas, cutting down trees, and thereby damaging both the habitat of many animal species and the main natural "filters" of our planet (i.e., the trees) [8]. Ultimately, deforestation and urbanization cause immense pollution.

Furthermore, this situation is exacerbated by the millions of cars that release carbon dioxide (CO2) into the atmosphere every day. Together with all the goods required to meet the needs of our modern society, these cars are manufactured in big factories that release additional pollutants into the environment and are progressively exhausting several sources of raw materials. In addition, to move people and goods, we consume millions of liters of fuel from oil exploitation - another limited resource that causes environmental pollution.

Those mentioned before are just some examples of the harm we are doing to our planet. Human actions and our consumption and production-related activities are affecting the health and functioning of our planet [60], and have significant consequences on the climate. Over time we have come to notice an increase in extreme temperatures associated with climate change. In other words, we have higher temperatures in summer and lower temperatures in winter [2]. Results of several studies based on long-time series data from the Northern Hemisphere reveal such an increase in the extreme temperatures and provide evidence of its relation with climate change [3].

This is a significant worldwide problem. However, in moderately evolved countries, like Portugal, it becomes even more worrying. These countries are not prepared for extreme temperatures, which results in an increase of the mortality rate [32] [68].

For example, in July 2006, a heatwave hit the city of Porto. In a study conducted to understand the consequences of this event on mortality (all causes) and morbidity (all causes, respiratory and circulatory diseases), it was observed that a one degree Celsius increase in mean apparent temperature would result in: a 2.7% (95% CI: 1.7–3.6%) increase in mortality (all causes), a 1.7% (95% CI: 0.6–2.9%) increase in respiratory morbidity a 2.2% (95% CI: 0.4–4.1%) increase in respiratory morbidity in women, a 5.4% (95% CI: 1.1–6.6%) increase in chronic obstructive pulmonary morbidity, and a 7.5% (95% CI: 1.3–14.1%) increase in chronic obstructive pulmonary morbidity in women, for the entire population. Moreover, considering only people above 75 years old, the study results showed: a 3.3% increase (95% CI: 1.7–5.0%) in respiratory morbidity, a 2.7% (95% CI: 0.4–5.1%) increase in respiratory morbidity in men, a 3.9% (95% CI: 1.6–6.3%) increase in respiratory morbidity in women, a 7.0% (95% CI: 1.1–13.2%) in chronic obstructive pulmonary disease, and a 9.0% (95% CI: 0.3–18.5%) in chronic obstructive pulmonary disease in women [57].

The effect of high temperature on mortality and morbidity can also be influenced by air pollution [43]. Increased ozone levels and particle concentration can aggravate the number of deaths and hospitalizations during an extreme heat climatic episode [67].

In Portugal, the city of Lisbon (south) is more affected by cold temperatures than Porto (north) [31]. The same happened in the rest of Europe. Moreover, the mortality rate during extreme cold events in the European Nordic countries is lower than in the countries of the South, like Portugal, Malta, and Spain. One degree decrease in temperature is associated with a 1,35% increase in the daily number of total natural deaths. Cold temperatures also result in significant morbidity, including respiratory tract infections, mental health problems, and exacerbations of existing musculoskeletal, cardiovascular and respiratory diseases [10]. Much debate remains around the determinants and prevention of excess mortality during cold weather. For example, in the case of the United Kingdom, a recent report shows that 22% of the excess of winter deaths in the UK can be attributed to poor thermal conditions in buildings [39]. However, the seasonal effect of cold weather on deaths is, at least historically, modifiable, for example, winter deaths in Wales and England have dropped from a 5-year average of $\sim$ 70000 per winter in the early 1950s to $\sim$ 27000 in the late 2000s [9]. Other authors have reported a general declining trend in excess winter mortality in the last century. Such a trend can be explained by improvements in infrastructure, technology, and public health, and it also has an important bearing on what can be expected with future changes in the climate [6] [33] [52].

1.1 Extreme weather events in Portugal

When considering the specific case of Portugal, we observe a high correlation between death rate and extreme weather events (i.e., cold and heat waves). Several factors can explain this. Among them, we must mention that most of the buildings are old and do not comply with the minimum requirements regarding thermal comfort conditions [65]. Overall, the Portuguese housing stock is outdated - 33% of the buildings data back to the post-war period and are in need of renovation [46] - and home heating and cooling systems are still rarely used. Such a situation partially results from a governmental policy of the 60s (which has been abolished only recently) that kept the rent prices very low. Therefore, "the rental revenues were not sufficient for house owners to perform the adequate maintenance of buildings. This has led to a strong reduction of the rental market and significant decay of the Portuguese housing stock quality. Portugal housing is found to be one of the most energy inefficient of all housing in the Europe Union" [46].

Building regulation incorporating energy performance considerations was adopted in Portugal in 1991, and its first remarkably successful application was on the buildings designed for the EXPO'98 housing stock. Yet, the buildings' energy certification system was introduced in Portugal only in 2006. The certification process aims to monitor and supervise the quality of new building projects and constructions, extensive renovations, as well as the status of existing buildings, especially when these are to be placed in the market for rental or sale.

The implementation of this process has implied creating a certificate database, which is managed by the Portuguese Agency for Energy (ADENE). Today the database contains information about over 1 million certified buildings of all ages [65].

Another aspect worth mentioning is the average income of the population. In Portugal, the average salary is lower compared to other European countries, and despite the cost of living being relatively low, the energy price is high. For this reason, people often avoid using electric heating or cooling devices, thus exposing themselves to unfavorable thermal conditions, or prefer to opt for cheaper solutions - like the use of fireplaces for heating since wood is more affordable than electricity. However, fireplaces release CO2 inside the house, exposing occupants to the risk of dying during the night due to CO2 inhalation [38].

Another factor is that the aging rate in Portugal is increasing (153,2% in 2018, against 98,8% in 2000) [44]. Older adults are more exposed to extreme temperatures because their immune system is less effective than young adults and are often affected by additional health conditions associated with aging. Moreover, many older adults live alone and therefore may experience difficulties in receiving assistance or get help in case of need [70] [64].

It is important to act quickly to mitigate climate change and its impacts. In order to do so, as pointed out in the UN 2030 Agenda for Sustainable Development, the following main actions should be undertaken: (1) Countries must take new preventive measures in order to minimize the damage caused by natural disasters (2) Integrate climate change measures into national policies, strategies and planning (3) Improve education, awareness-raising and human and institutional capacity on climate change mitigation, adaptation, impact reduction and early warning (4) Promote mechanisms to increase planning and management capacity related to climate change in developed and small countries [25].

Developing reliable risk indexes and early warning systems is fundamental to help reach the above-mentioned goals. These tools must include multiple data and indicators (i.e., based on weather forecast information as well as other variables - like socio-demographics indicators - that may influence health risk during extreme weather events). An early warning system based on a solid and reliable risk index is an effective tool that serves to (1) provide data and information that could be used for improving climate change mitigation policies, (2) raise public awareness on the correlation between health and climate change, and (3) communicate adaptation measures to the general public, thus strengthening its adaptive capacity. With these goals in mind, this thesis project aims at the development of a public geographic dashboard providing information on the buildings occupants' health risk during extreme weather events and is integrated into an FCT Research and Development project (RELIABLE), whose goal is to develop of a warning system that will extend and enhance the existing tools. Before describing the proposed solution, the following sections give an account of the main early warning systems used in Portugal and provide an overview of the RELIABLE project.

1.2 Problem Statement

Two of these systems are ICARO [68] and FRIESA. Are the main early warning systems for heat and cold events currently being used in Portugal to support public agents in implementing plans to minimize the health risks to the population during extreme weather events.

The ICARO was created in 1999 by Instituto Nacional de Saúde Dr. Ricardo Jorge (INSARJ), describes the impact on morbidity and mortality during heat waves in the five health regions of mainland Portugal. Every working day, between May and September, a bulletin reporting the risk level for each region and possible effects of the expected temperature variation on the population is shared with a list of health care institutions that are responsible for implementing mitigation measures.

Sixteen years later, in 2015, INSARJ developed FRIESA. Focused on monitoring extreme cold events during winter, FRIESA covers two districts (Lisbon and Porto) and estimates the mortality for all causes and respiratory and circulatory causes in both the general population and the population of 65 years old or above.

Based on the weather forecast, ICARO and FRIESA calculate a risk level, which different public agents then use to implement plans to minimize the health risks on the population (e.g., reinforcement of hospital staff during the cold or heat waves). However, the two systems are based on different risk indexes and differ in terms of spatial coverage and resolution, making the implementation of integrated strategies to mitigate the effects of extreme weather events less straightforward.

These two systems have three main flaws: (1) they do not target the general public - for instance, ICARO is directed to health care institutions, (2) they do not cover the entire national territory, ICARO covers five regions while FRIESA only covers the cities of Lisbon and Porto, and (3) are mainly based on weather forecast. As mentioned before, several other variables (e.g., buildings thermal condition) influence the population's health risk during extreme weather events. Thus, considering information like buildings thermal condition, occupant's age and sociodemographic indicators, is fundamental to develop a reliable risk indicator. Databases with such information do exist. For example, the Energy Certification of Buildings System (SCE) database provides information about the quality and thermal characterization of the residential building stock in mainland Portugal. Through the SCE database, we can access several certification parameters - climate and location, period of construction, and many others - related to the rating outcomes [65].

The project described here aims at addressing the above-mentioned issues by developing a public, geographic platform to communicate the health risks of buildings' occupants during extreme weather events. The system will have high spatial resolution and integrate data from multiple existing systems (ICARO and FRIESA) and databases, (i.e., census data and Portuguese buildings certificates database) in a single model.

By considering all these sources of information we will create a new risk indicator that will extend the existing ones in several ways. While ICARO [68] and FRIESA, are at the district level (eight in the mainland and 2 on the islands), the new indicator will have a higher spatial resolution - i.e., the subsection level. Specifically, the risk level will be calculated for each of the 178364 Portuguese 'statistical subsections' (census blocks). Moreover, the new indicator will improve the existing risk forecasting models by including new data sources like buildings characteristics and sociodemographic indicators. And finally, the new system will target not only experts and public agents but also the general public.

1.2.1 RELIABLE Project

This thesis is part of an FCT research project called RELIABLE (DSAIPA/DS/0111/2019), which falls under the UN Sustainable Development Goal nº13 ("Take urgent action to combat climate change and its impacts"). RELIABLE aims at developing an improved early warning system for health risks during extreme weather events. Specifically, RELIABLE aims at extending the current surveillance capabilities in the following ways:

- It improves the current risk forecasting models by integrating weather forecast information and other data sources using machine learning algorithms.
- It increases the spatial and temporal resolution of the warnings – i.e., the risk is calculated every six hours at the statistical subsection level (BGRI).

- It targets an extended public (from public authorities to citizens), will therefore be publicly available and easy to integrate into the websites of different public authorities.

The initial four sources of data used to develop the risk forecasting model are:

- Historical information regarding healthcare indicators during Extreme Weather Events EWE. It is crucial to improve the models that correlate the incidence of extreme weather events with (i) the type of events in the healthcare system (e.g., people seeking health treatment, deaths), (ii) demographics (e.g., elderly) and sociodemographic indicators (more exposed and vulnerable), and (iii) building stock characteristics (year of construction, typology). One of the core aspects is to assign the historical information to the BRGI level, thus improving the existing layers of information by distinguishing aspects like the exact location of the death (e.g., hospital), the actual residence (e.g., elderly peoples nursing homes) and the official/fiscal residence;
- Weather forecasts with high spatial resolution, where detailed information is calculated at the BRGI scale. The forecasting models will be improved taking into consideration the historical weather data and health data (mortality, morbidity and other indicators);
- The national census data of 2011, which needs to be updated (data of the last census will be available in 2022), particularly regarding the age and sociodemographic. This will be done by upgrading the National Statistics Institute (NSI) data with new data sources like digital information from municipalities (e.g., tax over estate/IMI information), satellite imagery and crowdsource information.
- The database of more than 1 million buildings Energy Performance Certificates (EPC) that were already issued by ADENE in order to estimate the indoor thermal comfort conditions during the EWE. This is the key source of information as it allows to estimate in detail the thermal comfort experience inside the buildings.

The municipality of Montalegre was selected as a testbed by the RELIABLE team. However, the model will eventually be extended to the whole Portuguese territory.

1.3 Proposed Solution

This thesis focuses on the development of the public geographic dashboard providing information on the buildings occupants' health risk during extreme weather events (EWE) and will represent the 'interface' for the models resulting from RELIABLE. The information provided through this platform needs to be updated every six hours (the minimum temporal resolution of the weather forecast models) and must be presented with a high spatial resolution (subsection level). Considering the goals of the RELIABLE project, several aspects should be considered in the design and development of the dashboard. In particular:

- the back-end architecture must be flexible so as to allow for the integration of additional data sources (which may also have different data structures) and, in the future, be extended to the entire national territory.
- the dashboard should be designed in such a way that it can be integrated into the websites of different public authorities. For this purpose, an Application Programming Interface (API) will be created.
- the front-end should be designed in a user-friendly way so as to ensure easy access to information also to non-expert users.

1.4 Document Structure

The document is divided into nine main sections. Initially, we start by briefly introducing the problem and the proposed solution to solve this same problem with the introduction section. Subsequently, a review is presented of what already exists in the current state of the art related with this problem.

After and based on conclusions obtained after carrying out the literature review, a more detailed solution to the problem found is presented. Here we have already discussed the application's architecture, among other aspects. After the solution was outlined, it was necessary to plan, as is common in all software developments, to have a plan to follow during the entire thesis execution time.

With this being said, we have the basics ready to start implementing the application, and it is in the Implementation section we show the details and the implementation decisions that were made so that the application meets our goals. Two types of tests were carried out to verify if the application reached the initial objectives: usability tests with five users and portability tests to verify if the application is easy to install in any environment.

Finally, the results of the previous tests are exposed. Based on these results, a discussion is made, and finally, we draw our conclusions based on all the work carried out.

2 Literature Review

This section provides an overview of the state-of-the-art and is divided into three principal subsections. The first one presents some existing risk communication dashboards (academic and commercial projects), all related to weather data, except one that provides information about the Covid-19 pandemic. The analysis of these platforms focuses either on the technology used or the layout structure and will inform the design of this thesis project.

The second section presents the mathematical models, which are the primary data sources for elaborating the model used in this project (described in the previous section). The third section proposes a review a set of technologies that could be used to develop the system proposed in this thesis.

Finally, the chapter ends with conclusions drawn from this analysis and a summary of the main design choices made to meet the project goals.

2.1 Platforms

This section presents the analysis of six existing risk communication platforms and ends with a review of scientific literature describing the main principles for designing such tools.

2.1.1 SusCity

This system materializes the Urban Building Database UBD user interface. The UBD is a database that integrates multiple information related to the urban built environment which comes from different sources and in diverse formats. The goal of an UBD is to provide data and information and facilitate knowledge creation and sharing on existing urban building stock and energy use to society. Its design and implementation must also consider complementary concepts and functionalities, which are all contributing to the overall goal of providing an integrated data repository that can be made available in a universal format for different users.

SusCity uses the UBD as a massive raw data storage for historical and real-time datasets to build decision models, reports and dashboards for the analysis and visualization of relevant city indicators. This dashboard allows for the interactive visualization of information related to the urban context and data exploration with a high spatial resolution.

Through the map displayed at the center of the interface, users can select one or multiple buildings, and visualize the related information available in the UBD, which is presented on the right side of the display. The left side presents the information summarized at urban scale (city, parish or selection area), allowing the dynamic multi-scale comparison of a specific parameter. In this way, decision-makers, owners or occupants can check their building's energy performance compared to their neighborhood's average, and identify opportunities to decrease energy consumption.

The city of Lisbon was selected as a testbed for this system. For such a purpose, a UBD containing data from 3259 buildings within 3.725.894 m^2 of built area [58] was created. Figure 1 shows the layout of the system.

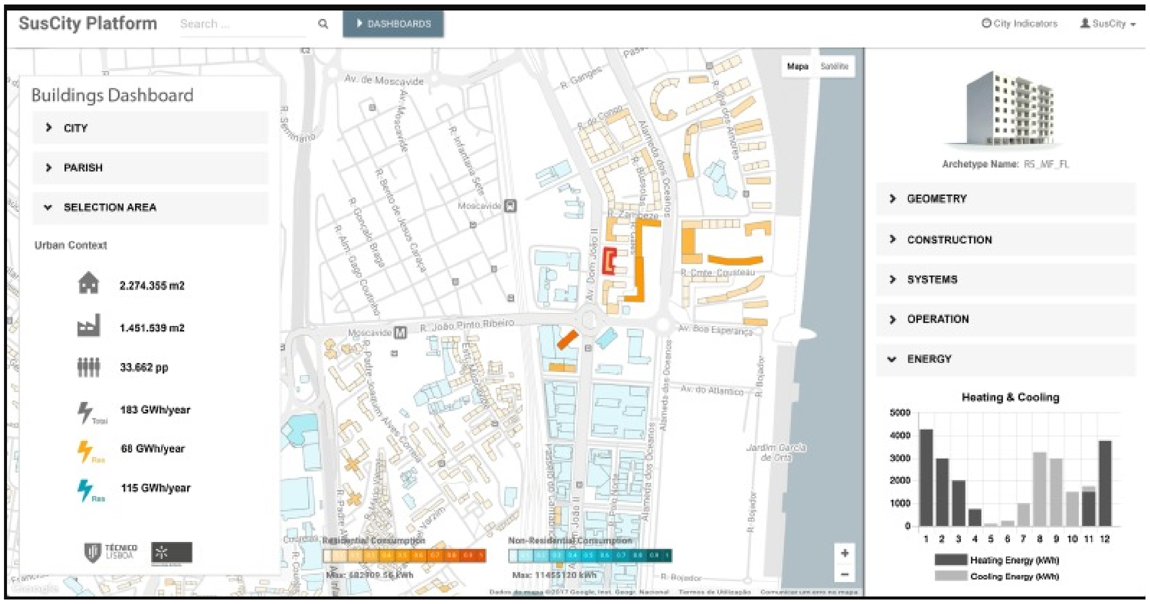


Fig. 1. SusCity Dashboard

2.1.2 ViewExposed

ViewExposed is an application aimed to help local stakeholders identifying areas that are particularly vulnerable to extreme natural events and understand what makes them vulnerable. A multiple view approach is adopted to visualize the information. Data is presented through a combination of parallel coordinates plot, table view, sparklines, and profile report [66].

Vulnerability assessment data is complex, so the authors implemented an easy-to-use interface facilitating a high degree of user interaction through multiple and linked views.

The integrated vulnerability assessment framework comprises four significant steps:

- Create vulnerability indices separately for storms (StoVI), floods (FloVI), and landslides (SliVI);
- Perform a weighted combination of the vulnerability indices for storms, floods, and landslides to generate a combined Physical Vulnerability Index (PhyVI);
- Assess the social vulnerability for Norway and create a Social Vulnerability Index (SoVI);
- Integrate physical vulnerability and social vulnerability indices into one, the Integrated Vulnerability Index (IntVI).

As shown in figure 2, the layout is divided into three main sections:

- **Map view:** Was implemented using the choropleth mapping technique and enriched the tool with spatial and thematic navigation functions [47].
- **The Parallel Coordinates View:** The authors decided to employ a parallel coordinates plot technique as broadly recognizable within the cartographic community due to its suitability for visual exploration of multivariate data. In addition, this technique is well suited for being linked with choropleth maps [30].
- **The Table View:** Displays the attribute information on the various vulnerability indices. The rows represent municipalities and the columns represent vulnerability scores.

2.1.3 VisAdapt

This solution is a visualization tool to support climate change adaptation [48]. It differs from current state-of-the-art visualization tools in climate change and adaptation in that

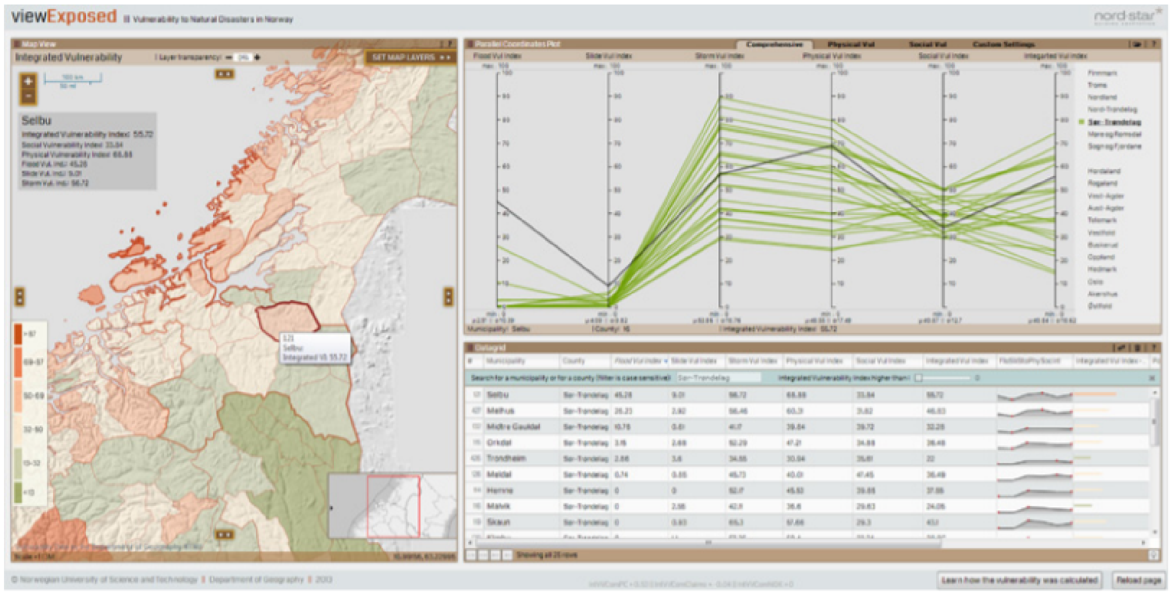

viewExposed
Vulnerability to Natural Disasters in Norway
Map View
Integrated Vulnerability
Selbu
Parallel Coordinates Plot
Datagrid

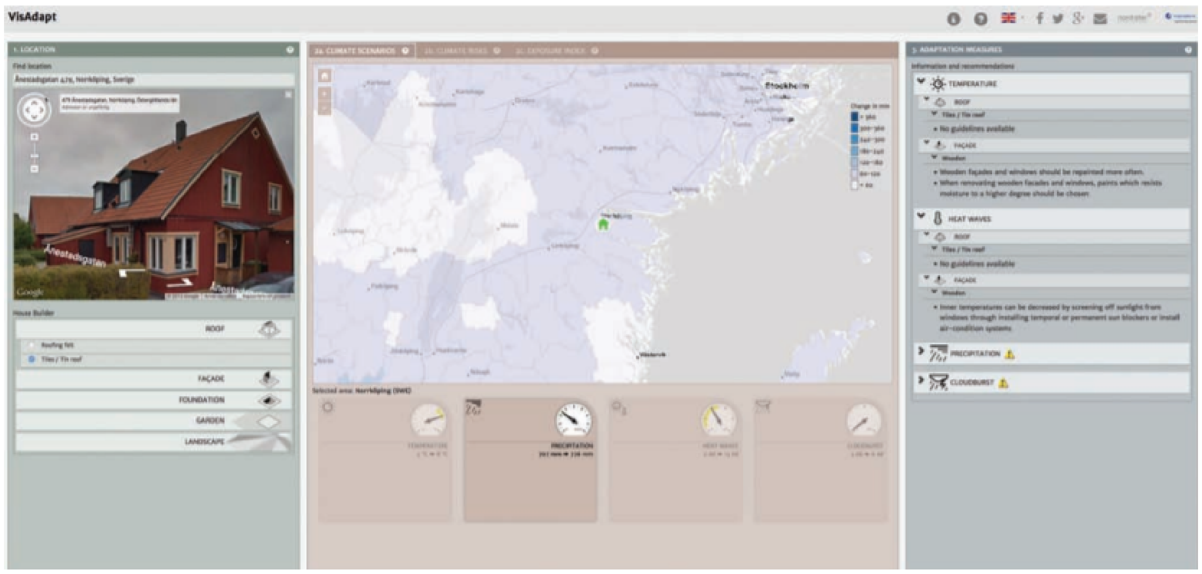
Fig. 2. The ViewExposed user interface

the authors developed it specifically for nonexperts users. The interface is structured as a three-step exploration process, from left to right, with three linked modules: from locate to investigate to act. In this way, the user's attention is organized in a sequence of pre-defined and logical steps that can be illustrated using a real user scenario (figure 3).

- **First Module:** The user starts by entering his/her address. Then, the user is expected to specify the different features of his/her house, such as type of roof, the type of garden and whether it has a cellar. At all times, it is possible to get help by clicking on the question mark in the top right corner.
- **Second Module:** The parameters from the first module are combined with a multitude of information on climate scenarios, climate risks, and exposure indices. The user starts browsing through different climate variables. With a quick look at the change gauges, the user can see that the projected change in the annual number of days with a heat wave and the projected change in the annual mean temperature are most prominent for the given region. The second module also includes a map that helps the user see if, based on the projected increase in annual precipitation

and the number of cloudburst events, he/she will need to consider making house renovations in the future.

- **Third Module:** Indicates possible vulnerabilities for that location based on the house characteristics (for example, a wooden façade is vulnerable because of the likely increase in both annual mean temperature and the annual number of days with a heat wave) and suggests adaptation measures. For example, could suggest the house owner pay attention to the type of paint is used and consider repainting more often. VisAdapt also provides the user with information regarding how to decrease indoor temperatures during the warmest season.



- No guidelines available
- Wooden façades and windows should be repainted more often.
- When renovating wooden facades and windows, paints which resists moisture to a higher degree should be chosen.
- No guidelines available
- Inner temperatures can be decreased by screening off sunlight from windows through installing temporal or permanent sun blockers or install air-condition systems.

Fig. 3. VisAdapt Web Application

2.1.4 IPMA Dashboard

The IPMA dashboard is very simple as it provides the weather forecast for Portugal and its archipelagos (Madeira and Azores). As shown in Figure 4, the interface is very similar to the information presented all days on the newscast.

It offers the user information regarding atmospheric and maritime conditions. By moving the mouse over the items displayed on the map, it opens a tooltip with a summary of the weather conditions for a given area. If the user clicks on it, it navigates to a page with more detailed information for the upcoming days and, especially, the next twenty-four hours (figure 5).

This system has some flaws; for example, the limited spatial resolution - the information is provided at the district level. In addition, the information is spread across different pages - i.e., the main dashboard and the detail pages -, while proving all relevant information on the main dashboard would facilitate understanding [13].

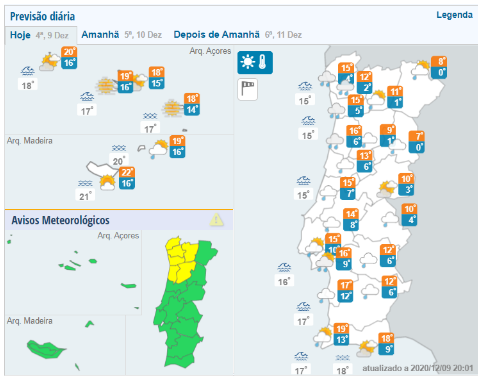
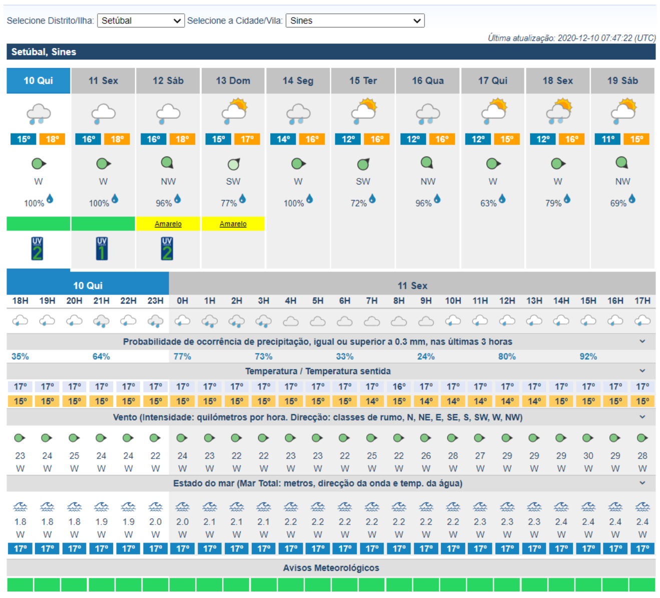


Fig. 4. IPMA Dashboard



Selecione Distrito/Ilha: Setúbal Selecione a Cidade/Vila: Sines

Última atualização: 2020-12-10 07:47:22 (UTC)

Setúbal, Sines

10 Qui	11 Sex	12 Sáb	13 Dom	14 Seg	15 Ter	16 Qua	17 Qui	18 Sex	19 Sáb
15º 18º	16º 18º	16º 18º	15º 17º	14º 16º	12º 16º	12º 16º	12º 15º	12º 16º	11º 15º
W	W	NW	SW	W	SW	NW	W	W	NW
100%	100%	96%	77%	100%	72%	96%	63%	79%	69%
🟩	🟩	Amarelo	Amarelo						
UV 2	UV 1	UV 2							

10 Qui						11 Sex																	
18H	19H	20H	21H	22H	23H	0H	1H	2H	3H	4H	5H	6H	7H	8H	9H	10H	11H	12H	13H	14H	15H	16H	17H
Probabilidade de ocorrência de precipitação, igual ou superior a 0.3 mm, nas últimas 3 horas																							
35%			64%			77%			73%			33%			24%			80%			92%		
Temperatura / Temperatura sentida																							
17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	16º	17º	17º	17º	17º	17º	17º	17º	17º	17º
15º	15º	15º	15º	15º	15º	15º	15º	15º	15º	15º	15º	15º	14º	15º	14º	14º	14º	14º	14º	15º	15º	15º	15º
Vento (Intensidade: quilómetros por hora. Direcção: classes de rumo, N, NE, E, SE, S, SW, W, NW)																							
23	24	25	24	24	22	24	23	22	22	23	23	22	25	22	26	28	27	29	29	29	30	29	28
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Estado do mar (Mar Total: metros, direcção da onda e temp. da água)																							
1.8	1.8	1.8	1.9	1.9	2.0	2.0	2.1	2.1	2.1	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.3	2.3	2.3	2.4	2.4	2.4	2.4
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º	17º
Avisos Meteorológicos																							
🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩

Fig. 5. District detail information

2.1.5 USA Severe Weather

The North American National Weather Service collects reports of severe weather events and weather-related damage the information is updated daily. The reports provided are preliminary, while official verified reports are stored at the National Centers for Environmental Information.

This dashboard (figure 6) displays reports from the one hundred twenty-two forecast offices across the United States. The data is automated and served in ArcGIS Online by the State of Georgia's Emergency Management and Homeland Security Agency [23].

The interface layout is divided into two main parts, the sidebar on the left provides a set of information about severe weather events in the USA: (1) reports on main severe weather conditions, (2) a bar chart showing the distribution of severe weather events across the different states and, (3) a list of these events.

The map displayed on the right of the screen presents the weather events listed on the sidebar through a set of icons.

The dashboard is highly interactive. By clicking on the elements presented on the map or the sidebar it is possible to filter the data or access additional information. For example, after selecting one of the states displayed in the bar chart, both the list of events and the map update to present information related to that state only.

2.1.6 DGS COVID-19 Dashboard

A more recent example of a risk communication dashboard is the one released on the DGS website, which relates to the Covid-19 pandemic [19].

The dashboard is presented in figure 7. The data is updated every twenty-four hours, and the time of the last update is always presented to the user.

The system is responsive, that is, it's available in both desktop and mobile versions. However, the mobile version does not load automatically when the user accesses the platform via a mobile device.

Despite the richness of data provided, the information is displayed in a clear and well-organized way. The dashboard is divided into four main parts which visualize the information in different ways.

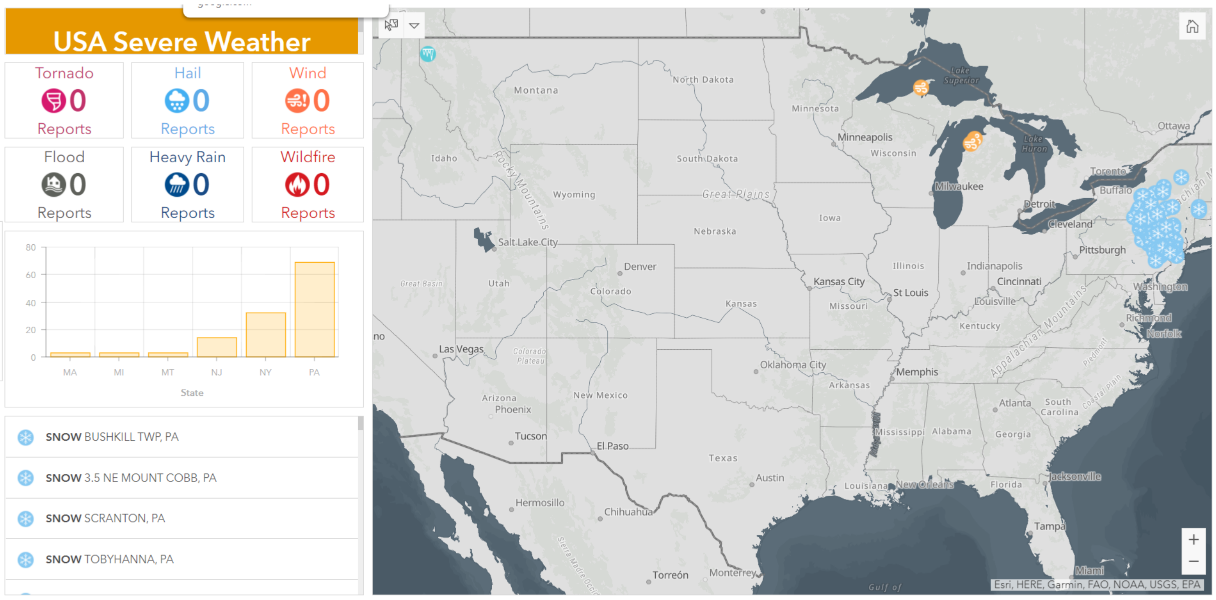


Fig. 6. USA Severe Weather Dashboard

2.1.7 *Map-Based Web Tools Supporting Climate Change Adaptation*

Neset and colleagues [61] analyze existing geo-visualization tools supporting climate change adaptation. In their article, the authors reviewed twenty map-based web tools. These tools are classified according to their content and functionality and assessed in terms of data presentation, interactive functions, information type, target audience, and the way aspects related to vulnerability and adaptation to climate change are addressed. Because climate change data are predominantly spatial, the authors pointed out that these tools frequently feature solutions typical for geographic visualization (geo-visualization), such as interactive maps or data displays [61].

Their study concludes that these tools can be classified into two main groups, depending on their functionality. The first group consists of the so-called data viewers, which allow users to display and view climate-related data. The second category is named data explorers. It includes those tools that offer more sophisticated functionality, allowing users to both display and analyze selected data, click on geographic objects, and view supportive graphs. Moreover, the authors observed that these tools normally feature moderate or high richness of data content. The majority of the systems analyzed target expert users

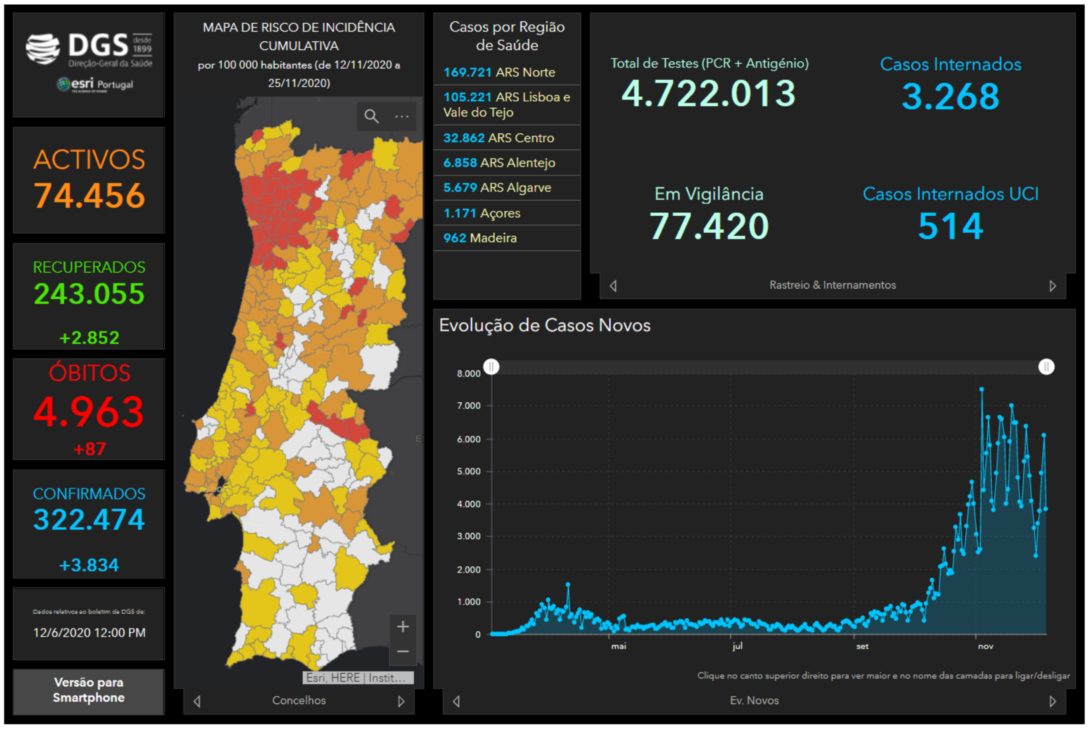


Fig. 7. DGS COVID-19 Dashboard

and are not intuitive, ultimately suggesting that incorporating input from user evaluations could greatly improve ease of use and understanding. According to Neset et al. [61], the general challenges when designing map-based tools supporting climate change adaptation appear to be (1) integrating information on adaptation measures with climate change– and impact-related information and (2) maintaining a balance between representing the inherent complexity and vast information, on one hand, and the fairly broad user profile of most tools, on the other.

2.1.8 Principles to create a good dashboard

In their article, Matheus et al. [55] present the results of a study aimed at understanding how to design dashboards that could create a sense of transparency and accountability. Through literature review and case studies analysis, the authors identified a set of benefits and risks of using dashboards in the public sector. Then they elaborated a set of ten design principles (see Table 1) that can lead to realizing the benefits and overcoming the risks identified [55].

Dashboard Principles		
Principle	Description	Source
Collect accuracy and precise data	Governments must give the most correct and precise information, to prevent users from being unable to understand the data and being misled. Incorrect information in the dashboard can result in bad decisions.	[53], Case Studies
Customize views	Dashboards should not be merely simple or generic visualizations; dashboards should contain customized views for showing the problem at hand. In this way decision makers and users can gain insight. Customized views can help them understand the situation. The design requires understanding of organizational strategies, viewpoints, business processes, indirect effects, decision support systems, and priorities. In the cases found in the literature review, separate apps are developed for each purpose to enable a clear view of the problem at hand.	[40], [49], [51],[16], Case Studies
Support different view	A single view might result in a limited picture on the situation. Different views can avoid bias and improve the understanding. By providing raw data, others can create new views which can result in updating the dashboard and improving usage.	[40], [49], [51],[16], Case Studies
Clear presentation	Dashboards enable the use of charts, graphs, pictograms, bars, and numbers, etc., to visualize information for monitoring and analyzing performance. Dashboards should visualize data in and easy-to-understand manner. In our cases, the simplicity of the dashboards enabled their use by a broad public.	[34], [42], [71], Case Studies
Offer decision-making support	Relationships between performance metrics and organizational desires must be clear. Dashboards can provide decision-support to evaluate ‘what if’ scenarios and to use predictive analytics. This can help provide more insight into the situation and help decision makers. By providing insight into possible alternatives, the effect of choosing an alternative can be predicted in our cases.	[45], [71], Case Studies
Interaction support	Static dashboards often provide limited insight. More insight can be gained by providing interaction features, which enables users to view the data from various perspective, to suggest recommendation based on the data but also to provide feedback to improve the use. Real-time information was a key element for supporting the decisions in the cases.	[5], [4], [56], [15], [7], Case Studies

2.2 Data Sources

To be able to carry out the proposed solution, it is necessary to have data. Here we will present some of these data sources that are used in RELIABLE project model.

2.2.1 Census Data 2011

The 2011 Census XV General Population Census and V General Housing Census has been made by National Institute of Statistics (NIS) with the collaboration of Municipalities and Parish Councils.

The Census is made every ten years in Portugal which records and systematizes data related to the population like age, gender, occupation. Individuals are counted within households and information is typically collected about the household structure and the housing, such as the floor area, the appliances ownership, and even the household maintenance needs.

Are essential elements for economic and social development, constituting themselves as indispensable instruments when including services and the definition of policies, in the most varied areas.

It provide useful information like:

- Which areas have the most degraded buildings;
- Which zones the population is older;
- The number of schools that are necessary;
- Where must be build the communication routes, hospitals etc;
- How to distribute the funds among the City Councils.

The territorial division is based on Nomenclature of Territorial Units for Statistical Purposes (NTUSP) whose structure is broken down by three levels:

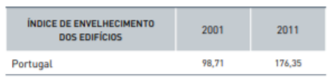
- NTUSP I: Covers Portugal Territory and Autonomous region of Madeira and Açores;
- NTUSP II: Covers North, Center, Lisbon, Vale do Tejo, Alentejo and Algarve;
- NTUSP III: Divide in Municipalities and Parish Councils.

The Portal of the National Statistical Institute discloses the main national results of the 2011 Census and is organized as follows [20]:

- Results Analysis;

- Quality Inquiry;
- Tables with the results up to the level of NUTS II;
- Methodology and Concepts.

For example, one of the study points of the census visible on 8 is the aging index of buildings:



ÍNDICE DE ENVELHECIMENTO DOS EDIFÍCIOS	2001	2011
Portugal	98,71	176,35

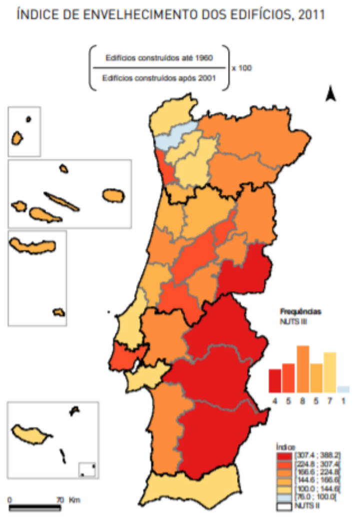
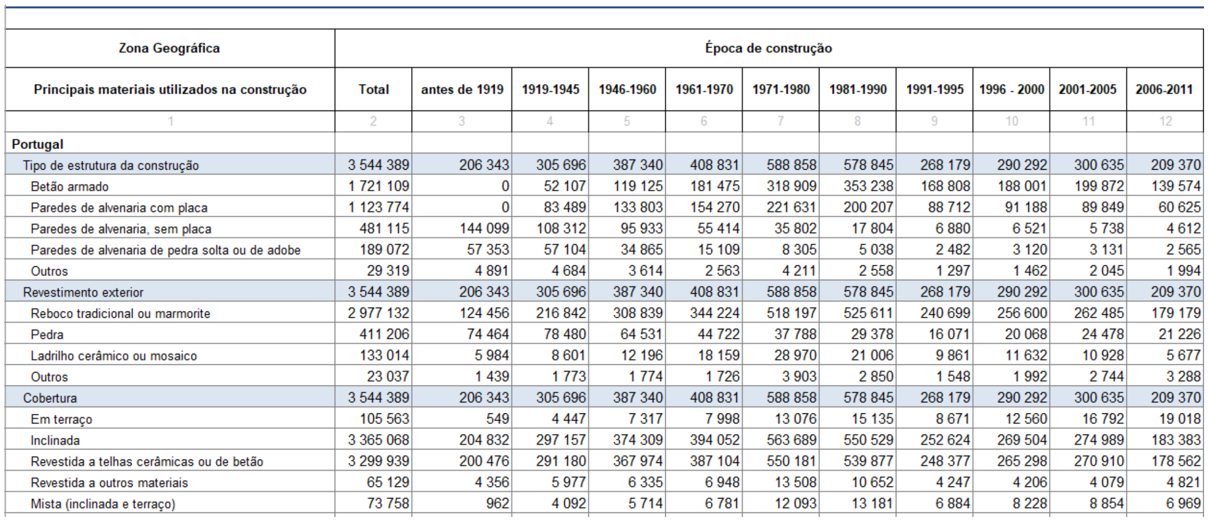


Fig. 8. Aging Index of Buildings

Another example is the about the information about materials used in construction (see figure 9):



Zona Geográfica	Época de construção										
Principais materiais utilizados na construção	Total	antes de 1919	1919-1945	1946-1960	1961-1970	1971-1980	1981-1990	1991-1995	1996 - 2000	2001-2005	2006-2011
1	2	3	4	5	6	7	8	9	10	11	12
Portugal											
Tipo de estrutura da construção	3 544 389	206 343	305 696	387 340	408 831	588 858	578 845	268 179	290 292	300 635	209 370
Betão armado	1 721 109	0	52 107	119 125	181 475	318 909	353 238	168 808	188 001	199 872	139 574
Paredes de alvenaria com placa	1 123 774	0	83 489	133 803	154 270	221 631	200 207	88 712	91 188	89 849	60 625
Paredes de alvenaria, sem placa	481 115	144 099	108 312	95 933	55 414	35 802	17 804	6 880	6 521	5 738	4 612
Paredes de alvenaria de pedra solta ou de adobe	189 072	57 353	57 104	34 865	15 109	8 305	5 038	2 482	3 120	3 131	2 565
Outros	29 319	4 891	4 684	3 614	2 563	4 211	2 558	1 297	1 462	2 045	1 994
Revestimento exterior	3 544 389	206 343	305 696	387 340	408 831	588 858	578 845	268 179	290 292	300 635	209 370
Reboco tradicional ou marmorite	2 977 132	124 456	216 842	308 839	344 224	518 197	525 611	240 699	256 600	262 485	179 179
Pedra	411 206	74 464	78 480	64 531	44 722	37 788	29 378	16 071	20 068	24 478	21 226
Ladrilho cerâmico ou mosaico	133 014	5 984	8 601	12 196	18 159	28 970	21 006	9 861	11 632	10 928	5 677
Outros	23 037	1 439	1 773	1 774	1 726	3 903	2 850	1 548	1 992	2 744	3 288
Cobertura	3 544 389	206 343	305 696	387 340	408 831	588 858	578 845	268 179	290 292	300 635	209 370
Em terraço	105 563	549	4 447	7 317	7 998	13 076	15 135	8 671	12 560	16 792	19 018
Inclinada	3 365 068	204 832	297 157	374 309	394 052	563 689	550 529	252 624	269 504	274 989	183 383
Revestida a telhas cerâmicas ou de betão	3 299 939	200 476	291 180	367 974	387 104	550 181	539 877	248 377	265 298	270 910	178 562
Revestida a outros materiais	65 129	4 356	5 977	6 335	6 948	13 508	10 652	4 247	4 206	4 079	4 821
Mista (inclinada e terraço)	73 758	962	4 092	5 714	6 781	12 093	13 181	6 884	8 228	8 854	6 969

Fig. 9. Buildings Construction Materials

2.2.2 ICARO

System created in 1999, has as main objective the detection of periods of heat that, based on the fulfillment of a set of criteria, are considered as involved on the mortality of the Portuguese population.

Daily between May and September, the ICARO system receives from IPMA, the day temperatures, observed and foreseen for all districts of Mainland Portugal. With which it is estimated the potential risk to the health of the population through ten Indexes-Alert-ICARUS, two national and eight regional (North Interior, North Coast, South Coast and South Interior).

Daily bulletins are issued to the competent authorities with risk information to current and next two days [68].

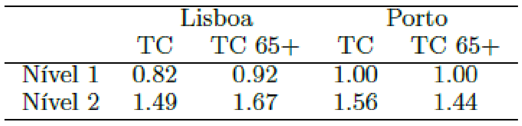
The the information is not centered in a platform to the entire Portuguese population, like a public dashboard. Another disadvantage is that the information is just about heat waves and not for cold waves too. And the data has a big spatial resolution, the district, which is imprecise.

2.2.3 FRIESA

FRIESA was developed by IPMA and INSARJ and is a surveillance and alert system of extreme cold for the districts of Lisbon and Porto. Aims to obtain a mathematical model capable of predicting risk situations for human health in extreme cold situations and to include it in a structured surveillance system for a pilot test.

The main product of the system is the FRIESA index, which is a daily risk measure that allows the identification of periods with possible impacts on mortality due to all causes or diseases of the circulatory or respiratory system due to extreme cold.

The FRIESA index is calculated to the general population and specifically to the population with more than sixty-five years old. In the next figure 10 the values of FRIESA index are presented to all-cause mortality from which they are expected effects of cold on mortality [69]. The mortality data come form the Daily Mortality Surveillance System (DMSS) [63].



	Lisboa		Porto	
	TC	TC 65+	TC	TC 65+
Nível 1	0.82	0.92	1.00	1.00
Nível 2	1.49	1.67	1.56	1.44

Fig. 10. FRIESA Index Levels

- **Index** > **Level 2:** Very probable effect on mortality;
- **Level 1** < **Index** <= **Level 2:** Probable effect on mortality;
- **0** < **Index** <= **Level 1:** Unlikely effect on mortality;
- **Index** = **0:** Null effect on mortality.

2.2.4 Buildings Energy Performance Certificates Data Base

Energy certification of buildings has only been in use in Portugal since 2006. The certification process aims at monitoring and supervising the quality not only of new building projects and constructions or big renovations, but also of existing buildings, particularly when these are to be put in the market for renting or selling.

In practical terms, the implementation of this process has implied the creation of a certificate's database, managed by ADENE.

Today that database contains information about over one million certified buildings of all ages. Given the number of details collected during the energy assessment, the database constitutes a great source of information allowing for a better picture of the current building stock.

Some of the parameters stored in this database are: climate zone and geographical location, age, type of construction and building physical characteristics [65].

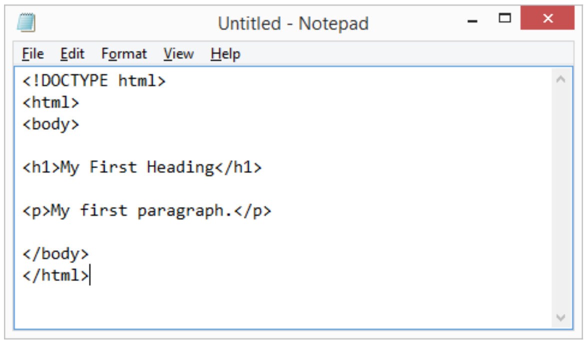
2.3 Tools Used

To build the dashboard of this thesis is necessary to study what technologies are best for achieving our goals. Here in this section, we will analyze some technologies used by applications reviewed in the Literature Review section and the ones we will use effectively to implement our solution.

2.3.1 Presentation

Here we will present some technologies used in the applications reviewed, some of them are standard in Web development for the presentation layer.

Hypertext Markup Language:
Hypertext Markup Language (HTML) is the standard language for web page development. Can combine it with two other languages, CSS and JavaScript.
Web browsers receive HTML documents and render the documents into multimedia web pages.
The page's structure to be shown to the user is entirely made using this language and it uses a tag system to create the elements to be displayed on the web page [28]. The next figure 11 show a very simple example of an HTML document:



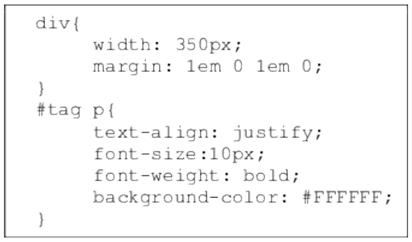
```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

Fig. 11. HTML Example

Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS) is a language that allows you to assign styles to documents written in HTML. It makes the separation between presentation and content it allows us to present the same content in different ways, in terms of layout, color and font. This brings some advantages such as flexibility [27].

The figure 12 shows a very simple example:

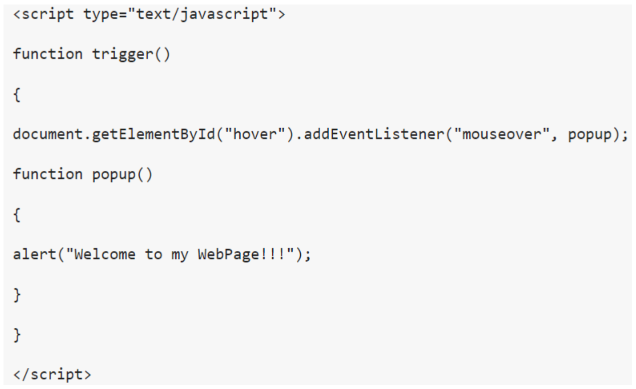


```
div{
     width: 350px;
     margin: 1em 0 1em 0;
}
#tag p{
     text-align: justify;
     font-size:10px;
     font-weight: bold;
     background-color: #FFFFFF;
}
```

Fig. 12. CSS Example

JavaScript/JQuery

It is another of the languages used in Web development. It is very efficient because it works asynchronously and allows us to manipulate HTML documents, perform actions on elements, handle events, communicate with the backend among other actions. It came to change the way Web development was done, and nowadays, it is actually one of the most used languages. [50]. A very simple example can be seen on figure 13.



```
<script type="text/javascript">

function trigger()

{

document.getElementById("hover").addEventListener("mouseover", popup);

function popup()

{

alert("Welcome to my WebPage!!!");

}

}

</script>
```

Fig. 13. JavaScript Example

Chart.js

Chart.js is a free open-source JavaScript library for data visualization, which supports eight chart types: bar, line, area, pie (doughnut), bubble, radar, polar, and scatter. Created by London-based web developer Nick Downie in 2013, now it is maintained by the community and is the second most popular JavaScript charting library on GitHub by the number of stars after D3.js, considered significantly easier to use though less customizable than the latter. Chart.js renders in HTML5 canvas and is widely covered as one of the best data visualization libraries [26]. The figure 14 shows an example of a simple bar chart:

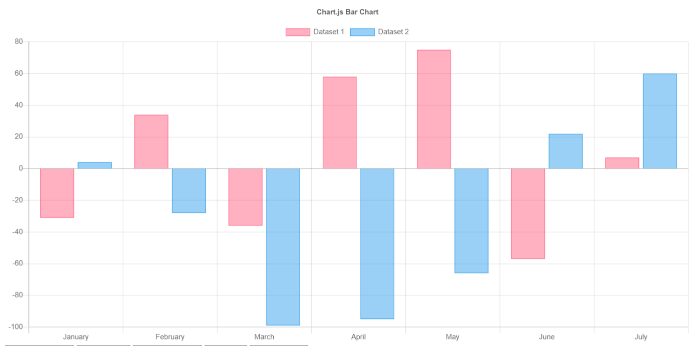


Fig. 14. Chart.js Bar Graph

Vue.js

Created by Evan You, Vue.js is an open-source progressive JavaScript framework for building user interfaces (UIs) and single-page applications, it is commonly referred to as Vue and it is currently one of the most emerging front-end technologies.
This framework uses "high decoupling", allowing developers to progressively create user interfaces (UIs) including more modules. The next figure 15 shows the libraries that we can include from a small Simple Page Application SPA to Multi-Page Applications MPA [54].

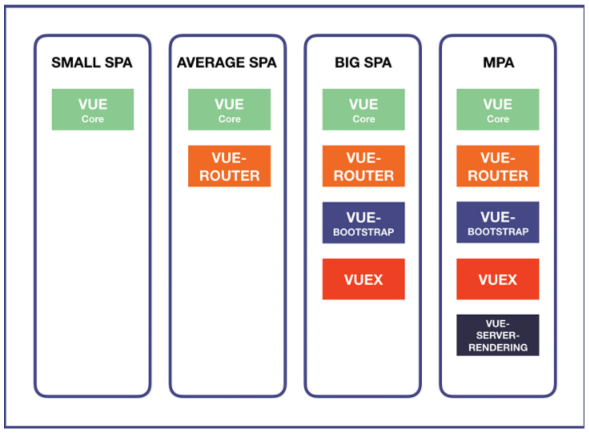


Fig. 15. Vue.Js Modules

Another of the main advantages is the use of components. What is a component? A component is basically a small part of the application, for example, a header, or a toolbar. In this way we can build large scale applications composed of small, self-contained, and reusable components. In the next figure 16 we can see and example of how a web application can be divided in components.

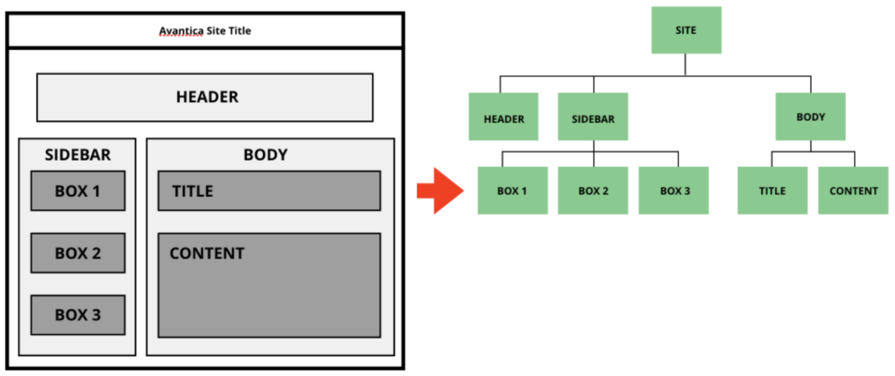


Fig. 16. Vue.Js Components

Some of Vue.js advantages are the simplicity, integration capacity, user-friendly interface, customizable, few restrictions, good documentation and large support.

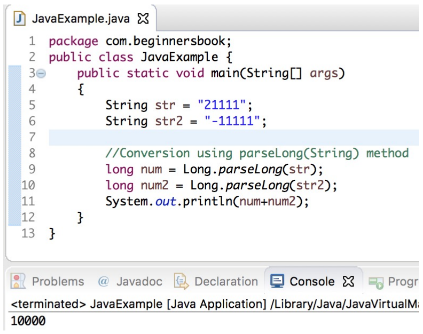
There are other well-known technologies, for example, the React and Angular but we will choose not to use them because they are not as flexible and simple to use compared to Vue.Js.

2.3.2 Logic

To be able to present data on the front-end, we need to have all the business logic mounted on the back-end, here we will review some technologies used to develop the back-end of some applications so that we can choose the best one for our particular case.

Java

Java is an object-oriented language designed so that the implementation has as few dependencies as possible. Two cases of its use are: (1) creating desktop applications or (2) developing services that other applications of various types will consume. It's designed for developers to create code once and run it anywhere. This means that once the code is compiled, it can run anywhere that supports Java without recompiling it again. [29]. In figure 17 is displayed a sum example:



```
package com.beginnersbook;
public class JavaExample {
    public static void main(String[] args)
    {
        String str = "21111";
        String str2 = "-11111";

        //Conversion using parseLong(String) method
        long num = Long.parseLong(str);
        long num2 = Long.parseLong(str2);
        System.out.println(num+num2);
    }
}
```

```
<terminated> JavaExample [Java Application] /Library/Java/JavaVirtualM
10000
```

Fig. 17. Java Example

Node.JS

It is a JavaScript-based framework that allows us to develop a complete web application. It is one of the most prolific Web development frameworks in 2019. It is used by big companies and small businesses alike.

Allows you to create scalable and flexible applications that process multiple connections at the same time. It works on a single task basis, that is, whereas traditional methods require a thread for each connection, Node.JS has a single thread for all connections. Can deal with a large number of requests without affecting response times. This comparison can be see on figure 18.

Where Node.JS really is good is to building fast, scalable network applications, as it's capable of handling a huge number of simultaneous connections with high throughput, which equates to high scalability.

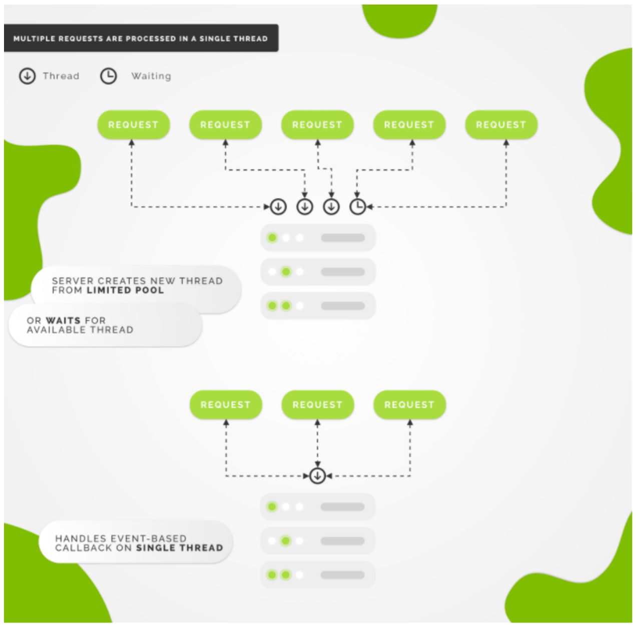

MULTIPLE REQUESTS ARE PROCESSED IN A SINGLE THREAD
Thread
Waiting
REQUEST
SERVER CREATES NEW THREAD FROM LIMITED POOL
OR WAITS FOR AVAILABLE THREAD
HANDLES EVENT-BASED CALLBACK ON SINGLE THREAD

Fig. 18. Node.JS connections

It works based on modularization, each application functionality represents a module, which means that the application is much better structured so that if we have to make any changes, we just have to change that specific module without affecting the others. Some of Node.JS advantages are scalability, flexibility, extensability, low cost, same language on backend and frontend and is the biggest world repository.

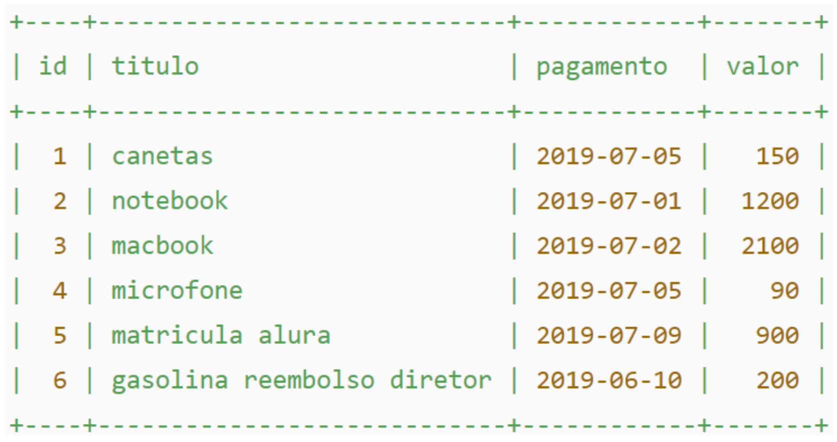
2.3.3 Databases

Another important part of an application is the data storage, to consume data from the data sources we need to have a way to save them. For this here we will present several database considered for our system.

Structured Query Language (SQL)

It is a standard language to work with relational data bases. Is a declarative language and does not require deep programming knowledge so that someone can write their queries and get their results. It is used in the main market's relational data bases like, Oracle, MySQL, MariaDB, Microsoft SQL Server [17].

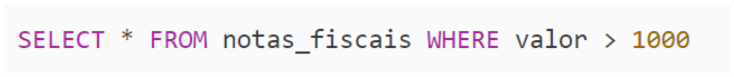
Imagine that we have the following database table 19 with invoice information and we want to get all the invoices with a value greater than 1000:



```
+----+---------------------------+------------+-------+
| id | titulo                    | pagamento  | valor |
+----+---------------------------+------------+-------+
|  1 | canetas                   | 2019-07-05 |   150 |
|  2 | notebook                  | 2019-07-01 |  1200 |
|  3 | macbook                   | 2019-07-02 |  2100 |
|  4 | microfone                 | 2019-07-05 |    90 |
|  5 | matricula alura           | 2019-07-09 |   900 |
|  6 | gasolina reembolso diretor | 2019-06-10 |   200 |
+----+---------------------------+------------+-------+
```

Fig. 19. Database table

With SQL we simple do the below query 20 and we will get the desired result:



```sql
SELECT * FROM notas_fiscais WHERE valor > 1000
```

Fig. 20. SQL Query Example

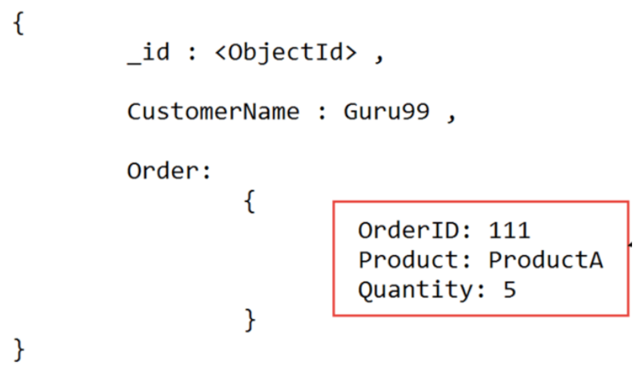
MongoDB

Unlike SQL, it is a non-relational database engine and is used for large amounts of data. Instead of using the traditional tables with relationships to each other, it uses collections and documents. Documents are equivalent to a set of data in key-value form. While the collection is equivalent to a database table in SQL, it groups a set of documents [24].
Some of the MongoDB features are:

- The database contains collections and these contain one or more documents. The size of the document depends on the number of fields, which means that not all of them are the same size.
- The structure of documents is related to the structure of classes in object-oriented programming.

- The documents (the rows) do not need to have a fixed structure previously created. We can add whatever structure is most convenient for us.
- The data model available within MongoDB allows you to represent hierarchical relationships, to store arrays, and other more complex structures more easily.

The MongoDB is very powerful when working with large quantity of data and when the structure of the data can change with frequency because it is very scalable.
In the next figure we can see an example of a document in MongoDb 21:



```
{
        _id : <ObjectId> ,

        CustomerName : Guru99 ,

        Order:
                {
                        OrderID: 111
                        Product: ProductA
                        Quantity: 5
                }
}
```

Fig. 21. MongoDB Document

2.3.4 Map Rendering

Based on the literature review many of the approaches used depend on maps in the presentation. We will soon review technologies that allow us to implement a map. The map is the key element in the dashboard, is a central element where is shown the main information to the user.

Based on this we will use this approach in our application as well.

Google Maps API V3

The Maps JavaScript API lets you customize maps with your own content and imagery for display on web pages and mobile devices. Its features four basic map types (roadmap, satellite, hybrid, and terrain) which you can modify using layers and styles, controls and events, and various services and libraries [18]. An example is displayed on figure 22:

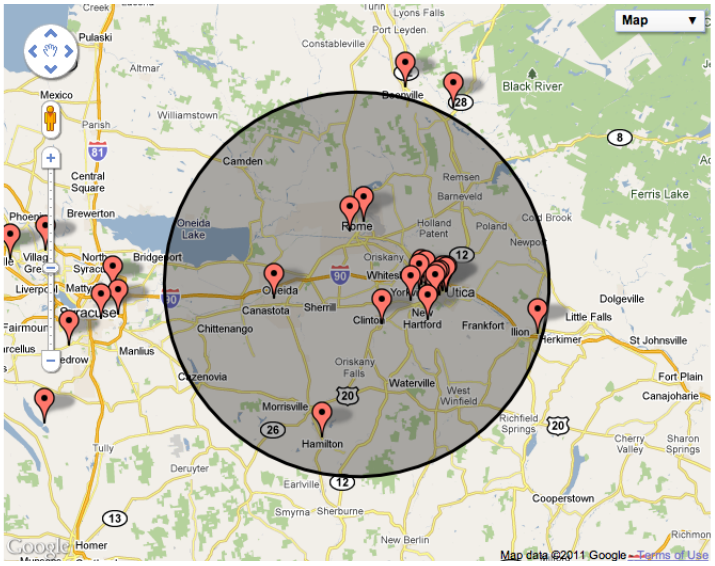


Fig. 22. Google Maps API V3

Leaflet

Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps. Weighing just about 39 KB of JavaScript, it has all the mapping features most developers ever need.

The leaflet is a library for interactive maps that is the leading mobile-compatible use. It is based on JavaScript. It was created to be simple to use, have high performance and usability. It works on several desktop and mobile platforms, has good documentation, has many plugins implemented to facilitate the developer's work [14]. In the next figure 23 we can see a basic example of how the map looks like.

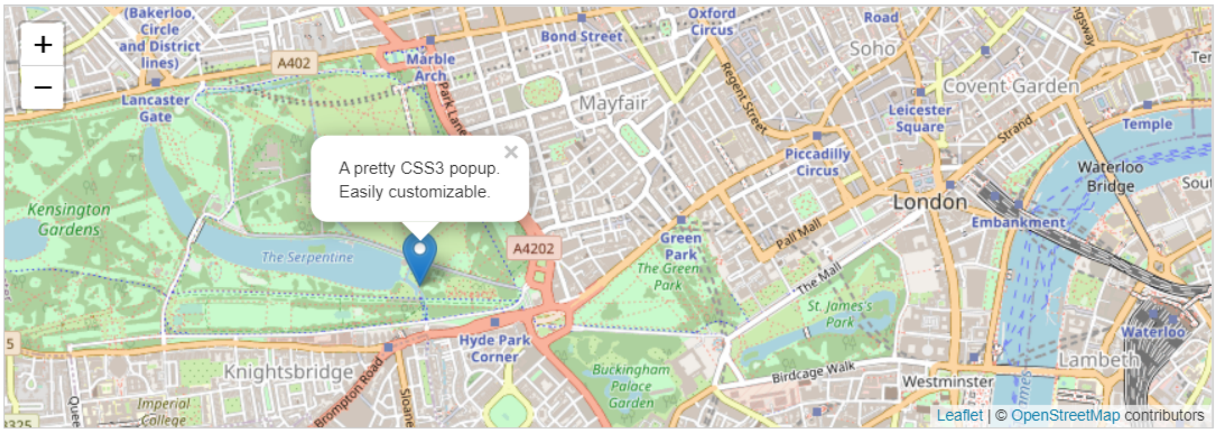


Fig. 23. Leaflet - Simple map example

Some of the main functionalities are, standard markers, custom markers with images, popups, is responsive on desktops and mobiles, zoom buttons, zoom with double click, scale in elements etc.

OpenStreetMap
It is a project to build a free geographic database of the world. It's aim is to eventually have a record of every single geographic feature on the planet. The database is populated thanks to contributors who collect GPS coordinates while driving or walking the streets. OpenStreetMap offers a wide range of maps that we can use in our applications or simply consult online. Several map rendering packages and Geographic Information Systems suport OpenStreetMap data, and all the software you need yo create maps is available free of charge, like for example Leaflet framework [36].

2.4 Summary

After all this work has been reviewed, we were able to draw some conclusions about what parts of each revised contribution we will use to achieve the objectives of this thesis.

Below we will organize these conclusions into three diffrente parts, (1) visualization layer, (2) Data sources and (3) Tools used to create our application.

2.4.1 Visualization

The organization of the elements visually is a fundamental factor for the user to understand the information that we are trying to convey to him in a clear and precise way, to do this, we will follow the next principles that we found on the literature review.

In all apps we noticed that all of them use a map, this element is the most common to show spatial information and continues to be the one that best performs this function, in our dashboard we want to show spatial information too, so we will use a map like the main element to show an overview of the information.

However, the map is not able to show all the information and with the detail we want, so we will use another influences that we found in this applications that we reviewed. For example, is very important show a detailed information about a certain event presented in the map. For this we will use different types of graphs and quick information indicators like for example the number of wind reports.

The use of flashy colors is a good approach to capture the user attention, so we will use this colors in the areas that we consider of most interest to the user.

Normally the information shown on a dashboard is complex and in a big scale, so it is important that the user has the ability to filter. Some of the dashboards reviewed use the standard filter but some of them use an interesting approach, the space on a dashboard is reduced so the filters are embedded in the dashboard elements themselves, this way we can save space and have a much more interactive filtering approach with the user.

Another aspect that is used is the quick information when the user places the mouse on top, this is a good point to use in solution of this thesis, because the user can access the information simply and quickly.

Just one of the reviewed application is not presented in a single page, here in our opinion a dashboard need to be single page so that the user can have all the information available without having to navigate between pages.

2.4.2 Data Sources

Data sources are crucial to the successful development of the dashboard in question. There are several sources of initial data for the thesis in question that could be expanded in the future.

Variables such as population age, number of daily deaths, thermal conditions of buildings, etc., are obtained through data sources such as censuses, FRIESA and ICARO systems, and the database containing energy certificates. These variables will not be used directly in the project, as they will be used in conjunction with the atmospheric variables by the members of the RELIABLE project to build the model that will give rise to the risk index.

On the other hand, we have the atmospheric variables consumed directly by the dashboard, for example, what temperature or wind speed is felt in a specific area. These data are obtained through an API created by the members of the RELIABLE project. This service receives a coordinate point and the meteorological variable and returns the meteorological values for that particular point.

For the above data to make sense, we have to associate them with the respective areas of the map. These zones are obtained through a JSON file that contains them, and initially, as already mentioned, they are zones only in the district of Montalegre.

With the combination of all these data sources, we have all the necessary data for elaborating our meteorological dashboard so that users stay informed about possible dangerous situations.

2.4.3 Tools Used

All the applications reviewed are Web Applications. Because of this influence, we will follow the literature in this aspect and create a Web Application. In this way, anyone can easily access the application.

To create an application with a good, responsive, extensible, and reusable interface, we will use the Vue.js framework. There are alternatives such as React and Angular, but for the purpose in question, we believe that Vue.js is more advisable because it is simpler and more expandable.

Another requirement was that the application needs to be scalable, flexible, and extensible because it was initially just for the Montalegre region, but in the future, it must be possible to expand to the entire national territory. To satisfy this requirement, we will use the Node.JS framework because it allows us to easily create a modularized application that can be easily expanded in the future.

Initially, we have the data sources as we described before, but the structure of these data sources can change. For example, for evaluating buildings' thermal conditions, new variables can be added, new data sources may emerge with the expansion to the entire Portuguese territory, and so on. To achieve these requirements we will choose to use a non-relational database, in this case, MongoDB because is very powerful when working with large quantity of data and when the structure of the data can change with frequency because it is very scalable.

Like we have seen in the 2.4.1 section we have two important elements in our dashboard, the map and the graphics. For the map we will use the Leaflet framework in conjunction with OpenStreetMap, this way we can take advantage of both, because is responsive across devices and allows us to make a series of customizations to the map. For the graphics we will use Chart.js because is simple to use and have a lot of chart types.

3 Solution

In this section we will present the solution for the problems that we previously stated in the Problem Statement section 1.2. Some of these problems are: (1) the existing systems are not available to the general public (2) the systems do not have a simple and comprehensive interface, some of that do not even have an interface (3) do not cover the entire Portuguese territory (4) and not have flexibility and the models used have few variables.

Taking into account the problems encountered, and in the conclusions that were drawn based on the literature review and our objectives, we want to create a public dashboard that can be easily used by general public with a easy and comprehensive interface, with more completed models (with more variables), which are flexible for new data sources or new variables in the existing ones, and extensible for all Portuguese territory.

3.1 Architecture

Based on this and in order to create the the system that meets our expectation we propose an architecture as presented in figure 24.

3.1.1 Application

The application is divided in two main parts and we will use the Node.Js framework to implement it.

The first part is the dashboard, that is the main output of this thesis. The user will use the dashboards to consult their data. And this is divided into three main parts, (1) the map where we will conjugate the Leaflet and OpenStreetMap frameworks and basically is where we will display the most of the information so that the user can access it quickly; (2) the interface, which aggregates everything that will be shown to the user in the application and in order to create a simple, responsive and comprehensive interface we will use the Vue.JS framework. And (3) the graphics, which will be used to show more detailed information about certain parts of the map, when for example an user clicks on a certain region in the map and will be implemented using the Graph.JS library.

The second part of the application is the back-end where we will handle all the data that comes from the data sources (database) in the data handler module which will be exposed through an API. The data handler module be prepared to read new data sources regardless of their format and store them in the database.

The API will be used to expose the data to different parts, like for example to our dashboard and to external applications that want to consume our data.

3.1.2 Database

The database will keep all the data from all the data sources introduced in the application. Here we will use MongoDB technology because as mentioned earlier, the application needs to be able to read any data source with any structure, so the MongoDB is a non-relational database and because of this it is more flexible than a relational database.

3.1.3 Notifications

The users will be able to configure certain locations in the application in order to receive alerts when something is wrong in these areas.

This is very useful, for example, a child who lives far from their parents can for example add the coordinates of the parent's house in order to receive alerts when something is not right in their area. In this way, the child can help the parents more quickly without having to constantly check the dashboard.

However, this is not the main objective of this thesis but one of the objectives of the RELIABLE project and will be implemented later by the project team.

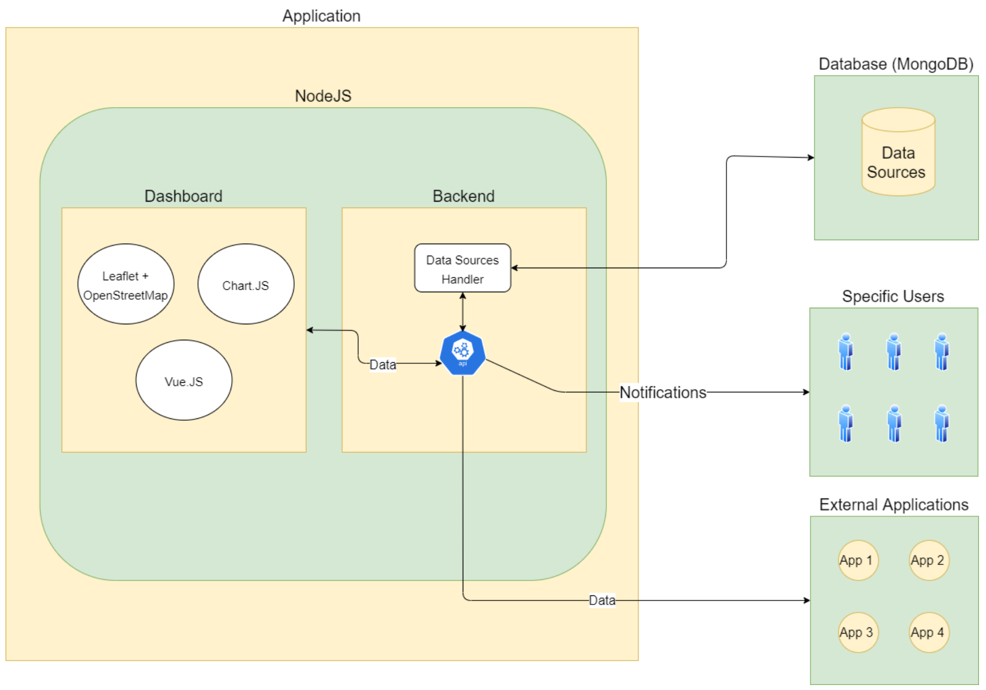


Fig. 24. System Architecture

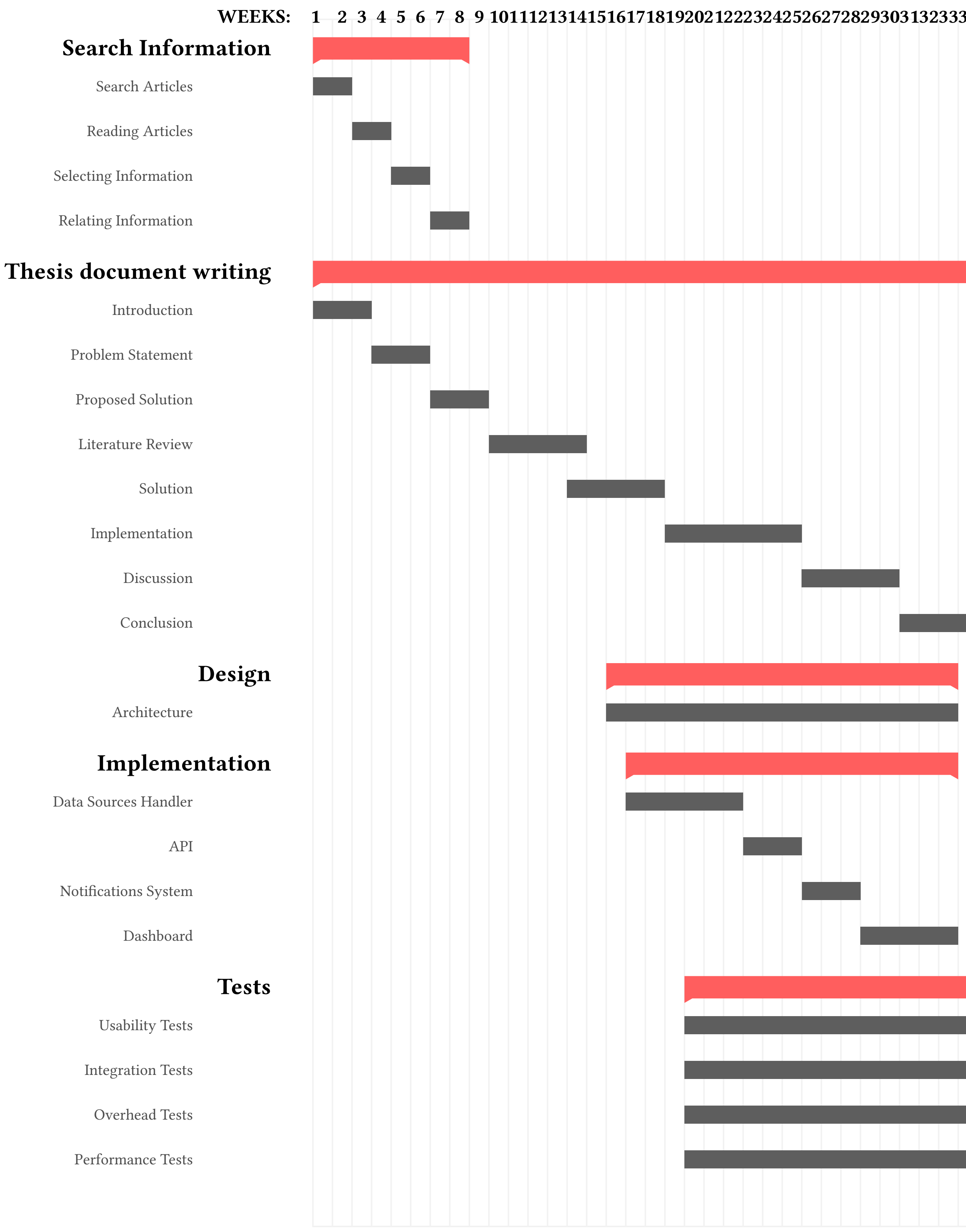
4 Planning

For the success of a project it is very important that an appropriate one be made in advance of all stages of the project. In this section we will present all the planning done for the execution of all the necessary tasks of this thesis. For this we will use the Gantt Diagram because it is the most used for this type of planning.

Some advantages of the Gantt Diagram are:

- More effective task segmentation;
- More transparent and effective distribution of responsibilities;
- Interdependence of activities;
- Clear definition of deadlines.

The week one corresponds to October 1, 2020 and the last week to June 30, 2021.


WEEKS: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
Search Information
Search Articles
Reading Articles
Selecting Information
Relating Information
Thesis document writing
Introduction
Problem Statement
Proposed Solution
Literature Review
Solution
Implementation
Discussion
Conclusion
Design
Architecture
Implementation
Data Sources Handler
API
Notifications System
Dashboard
Tests
Usability Tests
Integration Tests
Overhead Tests
Performance Tests

5 Implementation

In this section we will explain how the proposed solution was implemented to achieve the goals stated before.

To keep the application flexible and easily extended, we chose to use an architecture based on microservices. The architecture is divided in four microservices, (1) the Dashboard, (2) Data Sources Handler, (3) API and the (4) MongoDB Database.

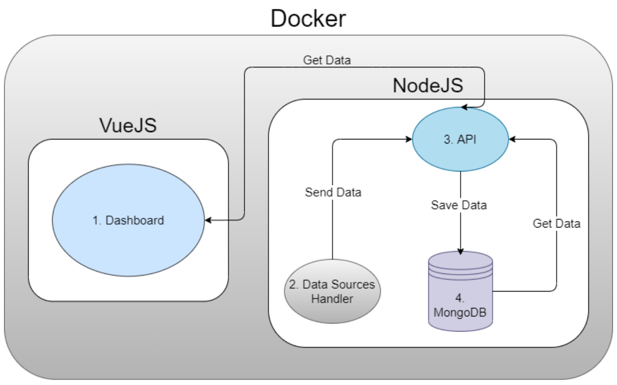
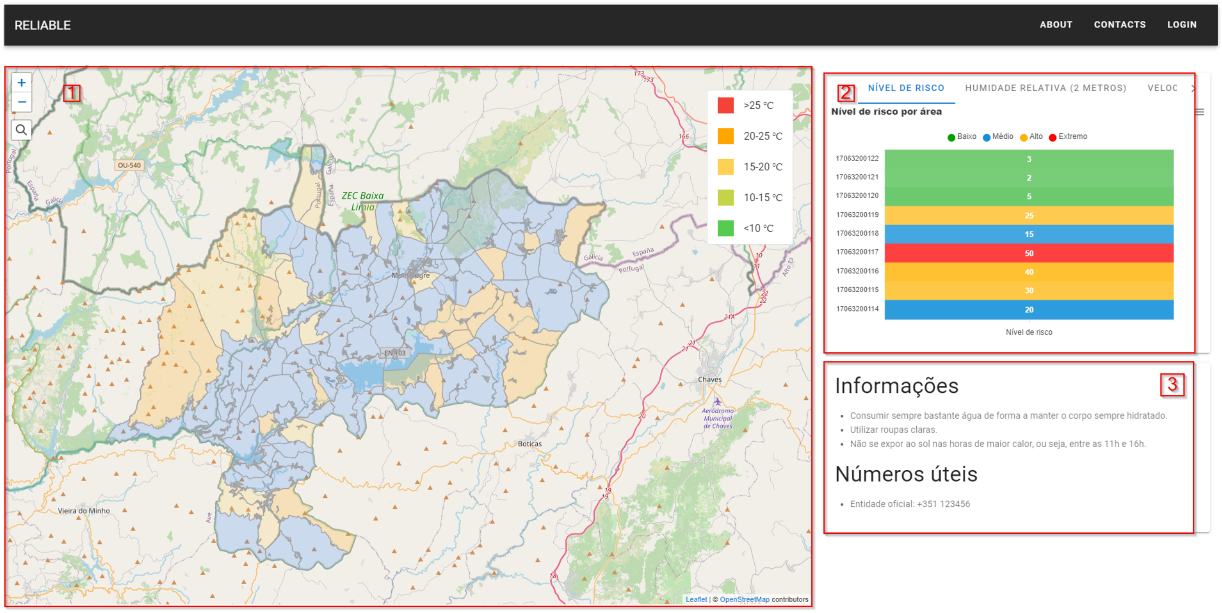


Fig. 25. Implementation Architecture

5.1 Dashboard

The Dashboard represents the main output of the application to users. It was implemented using Vue.js, HTML and Javascript and it is divided in three main components.



Informações

- Consumir sempre bastante água de forma a manter o corpo sempre hidratado.
- Utilizar roupas claras.
- Não se expor ao sol nas horas de maior calor, ou seja, entre as 11h e 16h.

Números úteis

- Entidade oficial: +351 123456

Fig. 26. Dashboard

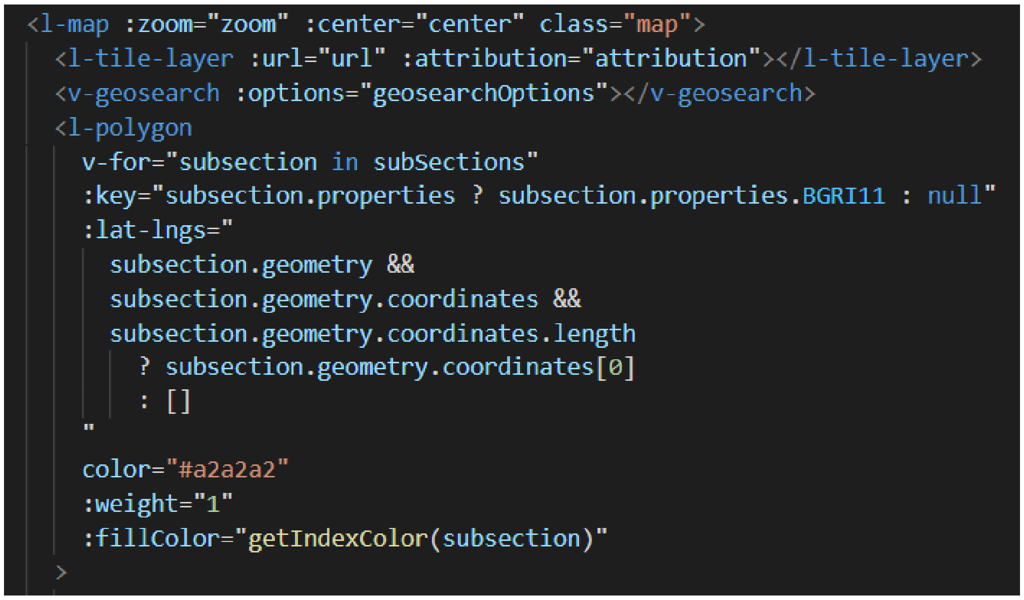
5.1.1 *Map View*

The map is the biggest and most important component in the dashboard. It is where the areas are drawn (so far we only have the zones referring to the municipality of Montalegre) and each zone has a corresponding color according to the risk index.

For the map, we are using the 'vue2-leaflet' npm library, which already has Vue components for map (l-map), polygons (l-polygon), markers (l-marker), and many others.

The 'l-map' component is simple. It just receives the zoom and the center coordinates. Inside this element, we have the 'l-polygon' component that represents an area inside the map, and it has two essential properties, the 'fillColor' that, as the name indicates, represents the area color. The second, 'lat-lngs', receives and arranges all polygon points to be drawn on the map (figure 27).

The implementation is flexible and allows us to use any variable in the color range. We are using the air temperature variable now, since the risk index indicator is not yet finished by the project partners.



```
<l-map :zoom="zoom" :center="center" class="map">
  <l-tile-layer :url="url" :attribution="attribution"></l-tile-layer>
  <v-geosearch :options="geosearchOptions"></v-geosearch>
  <l-polygon
    v-for="subsection in subSections"
    :key="subsection.properties ? subsection.properties.BGRI11 : null"
    :lat-lngs="
      subsection.geometry &&
      subsection.geometry.coordinates &&
      subsection.geometry.coordinates.length
        ? subsection.geometry.coordinates[0]
        : []
    "
    color="#a2a2a2"
    :weight="1"
    :fillColor="getIndexColor(subsection)"
  >
```

Fig. 27. Draw Map and Polygons

The map interface also allows the user to see more details about each area drawn on the map. When the user clicks on a map area, it shows a popup with detailed information. The information contains the ID, the name, and the air temperature, as we can see in the figure 28.

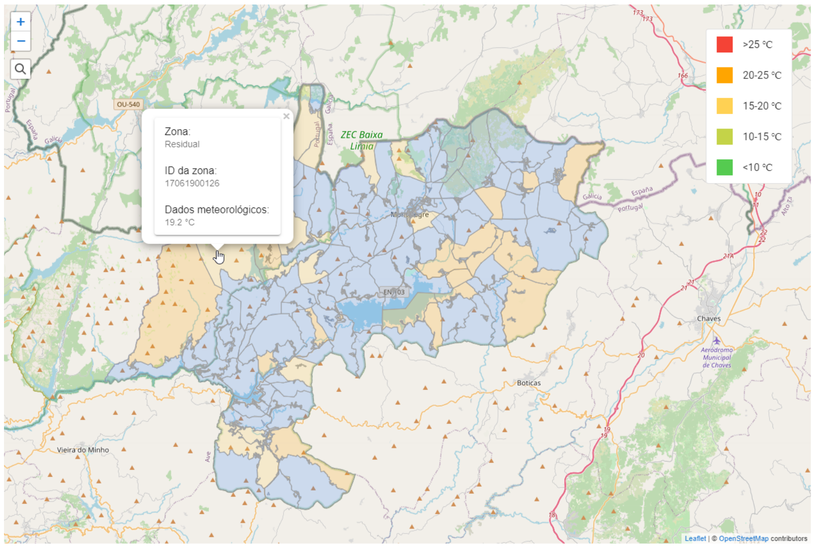
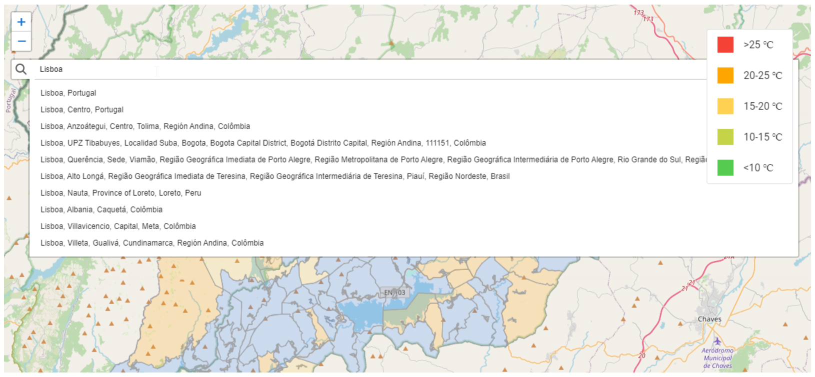


Fig. 28. Map Component

When the map expands for the rest of the Portuguese territory, we will have many more areas drawn on the map, making it difficult for the user to find the place he/she wants. With this in mind, we added a search field on the map to easily find the area they want (figure 29).


Lisboa

Lisboa, Portugal
Lisboa, Centro, Portugal
Lisboa, Anzoátegui, Centro, Tolima, Región Andina, Colômbia
Lisboa, UPZ Tibabuyes, Localidad Suba, Bogota, Bogota Capital District, Bogotá Distrito Capital, Región Andina, 111151, Colômbia
Lisboa, Querência, Sede, Viamão, Região Geográfica Imediata de Porto Alegre, Região Metropolitana de Porto Alegre, Região Geográfica Intermediária de Porto Alegre, Rio Grande do Sul, Regiã
Lisboa, Alto Longá, Região Geográfica Imediata de Teresina, Região Geográfica Intermediária de Teresina, Piauí, Região Nordeste, Brasil
Lisboa, Nauta, Province of Loreto, Loreto, Peru
Lisboa, Albania, Caquetá, Colômbia
Lisboa, Villavicencio, Capital, Meta, Colômbia
Lisboa, Villeta, Gualivá, Cundinamarca, Región Andina, Colômbia

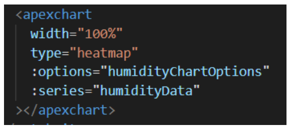
Fig. 29. Map Component

5.1.2 Detail View

At the upper right corner, as presented in figure 26, the detail area presents additional weather information for each of the areas marked on the map.

Motivated by the Literature Review, we decided to use the Chart.js library to build graphs in the Detail View. But when it came time to implement this feature and after more specific research, we decided to use the APEXCHARTS library because it is more flexible and straightforward to operate with Vue.js.

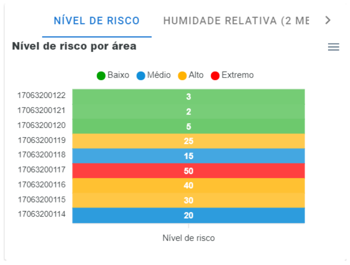
Figure 30 shows how we can create a chart with this library. We need to pass three properties, (1) type of the graph, in our case, we are using Heat Map type. (2) Series property is a simple array of values. And (3) options property consists in the graph configurations, like size, color ranges and many others.



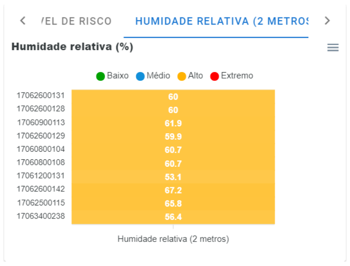
```
<apexchart
  width="100%"
  type="heatmap"
  :options="humidityChartOptions"
  :series="humidityData"
></apexchart>
```

Fig. 30. Chart Example

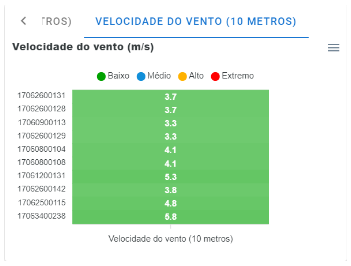
Here we have a card with three tabs, each one displaying a heat map with a specific weather variable type. For example, in the first tab we have information regarding risk index (as mentioned above, due to the fact that we still do not have access to risk data there is presented is dummy, figure 31a), second information about relative humidity (figure 31b) and finally in the third tab presents data regarding wind speed (figure 31c).



(a) Risk Index



(b) Relative Humidity



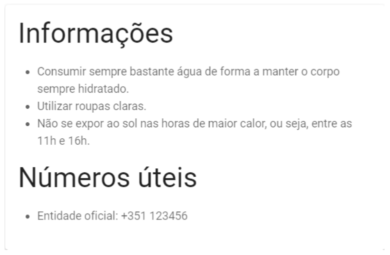
(c) Wind Speed

Fig. 31. Detail View

5.1.3 Information View

The information view is quite simple, it contains several suggestions that users should consider depending on the weather conditions, and it also presents some useful contacts to get more detailed information.

At this moment, the information is static. It will have tips for when tips for particular and important weather events, for example when its too cold or hot. This page will also present contacts for entities responsible for providing support to users for more specific questions.


Informações

- Consumir sempre bastante água de forma a manter o corpo sempre hidratado.
- Utilizar roupas claras.
- Não se expor ao sol nas horas de maior calor, ou seja, entre as 11h e 16h.

Números úteis

- Entidade oficial: +351 123456

Fig. 32. Information View

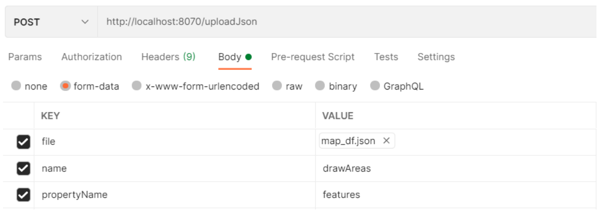
5.2 Data Sources Handler

This microservice is responsible to handle the 'static' data sources, such as an Excel file.

In this case, two different types of data sources handlers have been implemented, for Excel files and JSON files.

With this service, a system administrator can easily upload an Excel or JSON file to the server. This microservice sends an HTTP request to API that communicates with the database to insert the file data. The data will be placed in a new collection or in an existing one depending if the collection name already exists. An example of these data sources is the data referring to the zones to be drawn on the map, which is a simple JSON file.

In figure 33 we can see an example of a JSON file upload. For this request we have three possible parameters, the file itself, the name of the collection and the name of the JSON property that contains the data to be inserted, if we do not pass a property name, we will insert all the content from the JSON file in the database.


POST http://localhost:8070/uploadJson

Body: form-data

	KEY	VALUE
✓	file	map_df.json
✓	name	drawAreas
✓	propertyName	features

Fig. 33. Upload JSON File

5.3 API

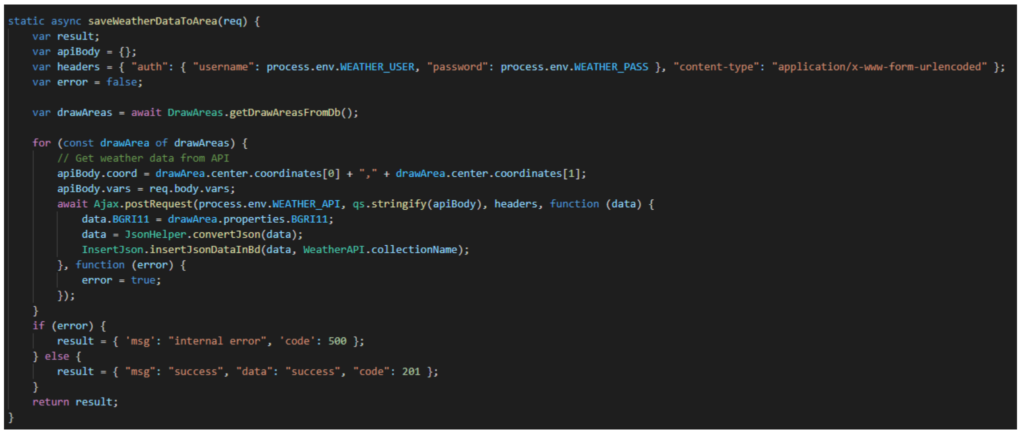
The API is the main component of the application. As the name implies, this microservice works as a data provider so the information can be presented in the Dashboard. All the information that our API sends to the Dashboard comes from our database. The API work as an abstraction of the database in order we can manipulate the data the way we want. Bellow we explain the main services provided by the API.

5.3.1 *Obtain Weather Data*

This is a background process, where our API obtains data from an external API provided by RELIABLE project managers and saves this data in our database. Its importante to clarify that when the Dashboard needs to consume data, the request is always made to our API, which will get the data from our database. The external API is never consumed at runtime, all the data used in the dashboard was previously written to our database.

Their API returns multiple metrics related to weather such as, air temperature, relative humidity, wind speed, and many others, for a specific location (identified by its coordinates). The weather forecast is made for the next seven days following the current day and predicts each hour of each day.

At this moment, we have a route prepared to be called by a cron job to run this background process every seven days, it is a simple method that for each polygon makes an HTTP request to the Weather API and obtains the weather forecast for the specified variable (air temperature, relative humidity...), like in figure 34. But as we know, a forecast can change over time, so it would be interesting to carry out this process every day or even hourly since the prediction is made for each hour of the day. To accomplish this is necessary to add additional logic so that if a forecast record already exists for a certain time and zone it is replaced with the most recent forecast.



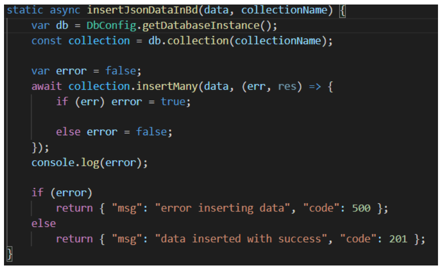
```javascript
static async saveWeatherDataToArea(req) {
    var result;
    var apiBody = {};
    var headers = { "auth": { "username": process.env.WEATHER_USER, "password": process.env.WEATHER_PASS }, "content-type": "application/x-www-form-urlencoded" };
    var error = false;

    var drawAreas = await DrawAreas.getDrawAreasFromDb();

    for (const drawArea of drawAreas) {
        // Get weather data from API
        apiBody.coord = drawArea.center.coordinates[0] + "," + drawArea.center.coordinates[1];
        apiBody.vars = req.body.vars;
        await Ajax.postRequest(process.env.WEATHER_API, qs.stringify(apiBody), headers, function (data) {
            data.BGRI11 = drawArea.properties.BGRI11;
            data = JsonHelper.convertJson(data);
            InsertJson.insertJsonDataInBd(data, WeatherAPI.collectionName);
        }, function (error) {
            error = true;
        });
    }
    if (error) {
        result = { 'msg': "internal error", 'code': 500 };
    } else {
        result = { "msg": "success", "data": "success", "code": 201 };
    }
    return result;
}
```

Fig. 34. Get Weather Data from API

In this thesis, we are using three different types of weather data: (1) air temperature (t2) showed in the map area popup, (2) relative humidity (rh2) in the graph of the detail view's second tab, and (3) wind speed (ws10) in the third tab. This data is stored in the 'weatherAPI' collection figure 40.

5.3.2 Store Data

All the data we want to insert in the database passes by the API. As we are using MongoDB engine for the database, all data is in JSON format, so we have a generic method that inserts any data formatted in JSON in a specified collection of the database (figure 35). In the future it would be a good idea to implement a routine inside the API to eliminate duplicate records within the same collection.



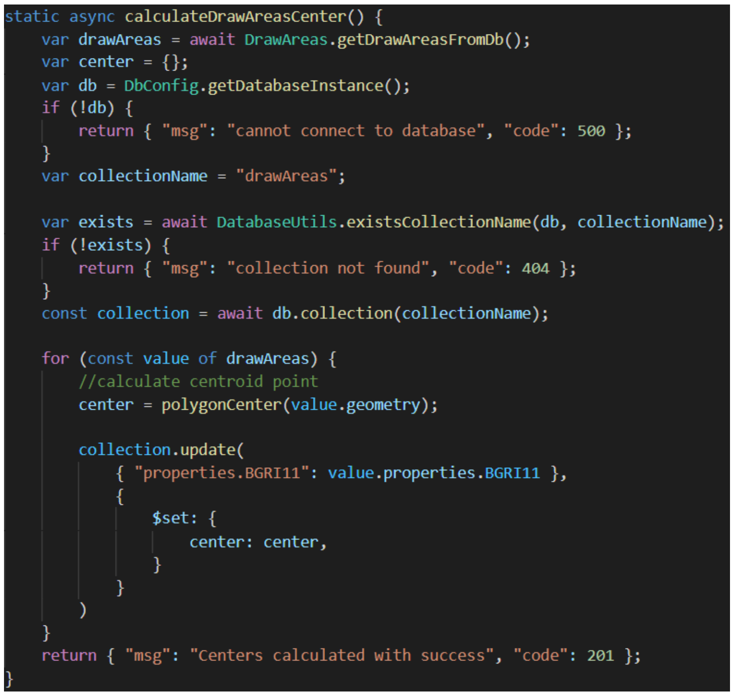
```javascript
static async insertJsonDataInBd(data, collectionName) {
    var db = DbConfig.getDatabaseInstance();
    const collection = db.collection(collectionName);

    var error = false;
    await collection.insertMany(data, (err, res) => {
        if (err) error = true;

        else error = false;
    });
    console.log(error);

    if (error)
        return { "msg": "error inserting data", "code": 500 };
    else
        return { "msg": "data inserted with success", "code": 201 };
}
```

Fig. 35. Store JSON Data

5.3.3 Map Areas

Each polygon drawn in the map has multiple coordinates points. So, for each polygon, we calculate the center point using the npm library 'geojson-polygon-center'. For this, we obtain all the records in 'drawAreas' collection. We calculate the polygon center point calling the 'polygonCenter' function of the library that receives all the coordinates points of the polygon and returns the center point for each of them. After calculating the point, we update the property 'center' with the center point coordinates in the corresponding collection element.

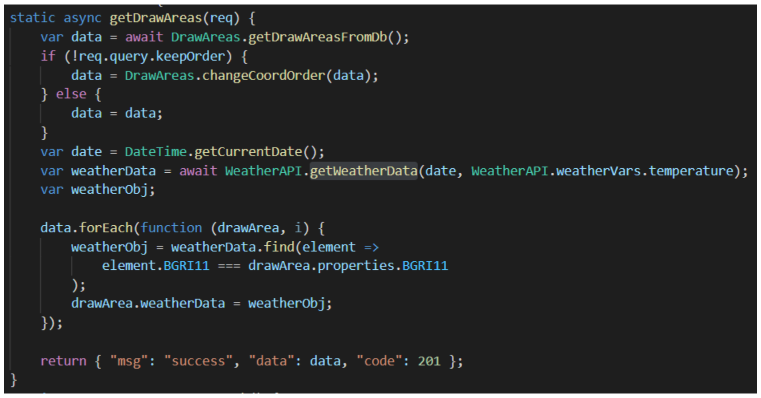
The code with this logic is present in figure 36.



```
static async calculateDrawAreasCenter() {
    var drawAreas = await DrawAreas.getDrawAreasFromDb();
    var center = {};
    var db = DbConfig.getDatabaseInstance();
    if (!db) {
        return { "msg": "cannot connect to database", "code": 500 };
    }
    var collectionName = "drawAreas";

    var exists = await DatabaseUtils.existsCollectionName(db, collectionName);
    if (!exists) {
        return { "msg": "collection not found", "code": 404 };
    }
    const collection = await db.collection(collectionName);

    for (const value of drawAreas) {
        //calculate centroid point
        center = polygonCenter(value.geometry);

        collection.update(
            { "properties.BGRI11": value.properties.BGRI11 },
            {
                $set: {
                    center: center,
                }
            }
        )
    }
    return { "msg": "Centers calculated with success", "code": 201 };
}
```

Fig. 36. Calculate Polygon Center

After the centroid is calculated we have a simple route that returns the map areas coordinates to be drawn in the map, and the current air temperature for each area. Two queries are made, one to get the map areas and the other to get the weather data separately, and after this, we map each area to the corresponding air temperature by the centroid point. All the information on this route is shown in the Map Detail View of the Dashboard. The route implementation is visible at figure 37.



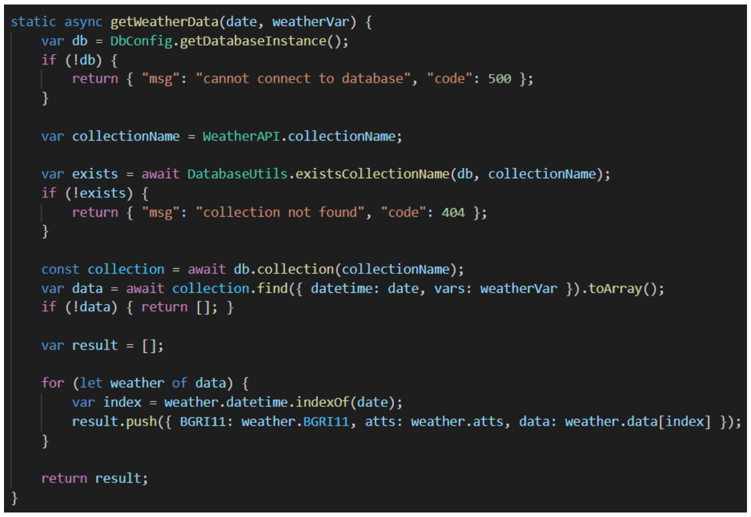
```
static async getDrawAreas(req) {
    var data = await DrawAreas.getDrawAreasFromDb();
    if (!req.query.keepOrder) {
        data = DrawAreas.changeCoordOrder(data);
    } else {
        data = data;
    }
    var date = DateTime.getCurrentDate();
    var weatherData = await WeatherAPI.getWeatherData(date, WeatherAPI.weatherVars.temperature);
    var weatherObj;

    data.forEach(function (drawArea, i) {
        weatherObj = weatherData.find(element =>
            element.BGRI11 === drawArea.properties.BGRI11
        );
        drawArea.weatherData = weatherObj;
    });

    return { "msg": "success", "data": data, "code": 201 };
}
```

Fig. 37. Get Map Areas and Air Temperature

5.3.4 Weather Data

After we obtained and stored the Weather data from the external API in our database, it is available to be consumed by Dashboard.

The Detail View of the Dashboard uses the route of figure 38 to show the different types of weather data in the various graphs. In this case, we are consuming the relative humidity and the wind speed, but we have many others variables available.



```
static async getWeatherData(date, weatherVar) {
    var db = DbConfig.getDatabaseInstance();
    if (!db) {
        return { "msg": "cannot connect to database", "code": 500 };
    }

    var collectionName = WeatherAPI.collectionName;

    var exists = await DatabaseUtils.existsCollectionName(db, collectionName);
    if (!exists) {
        return { "msg": "collection not found", "code": 404 };
    }

    const collection = await db.collection(collectionName);
    var data = await collection.find({ datetime: date, vars: weatherVar }).toArray();
    if (!data) { return []; }

    var result = [];

    for (let weather of data) {
        var index = weather.datetime.indexOf(date);
        result.push({ BGRI11: weather.BGRI11, atts: weather.atts, data: weather.data[index] });
    }

    return result;
}
```

Fig. 38. Get Different Types of Weather Data

5.4 Database

Since there is the possibility of adding new data sources and also due to the fact that the data structure can change with the addition of, for example, one more weather variable the database needs to be flexible, therefore, like mentioned before, we decided to use the MongoDB engine to the database.

We have a Docker Container running a Mongo image with the project database based on a volume, that is, even if the container stops the data that is saved on the database, this one is not lost. The database is divided into two collections:

5.4.1 DrawAreas

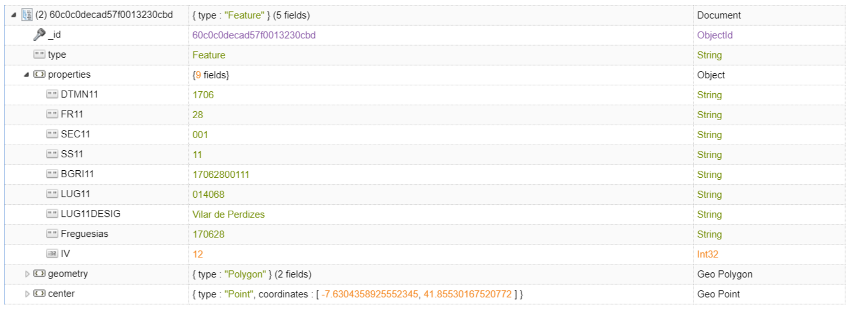
This collection contains the data referring to the areas to be marked on the map for the district of Montalegre. This data comes from a JSON file, and the data is inserted in the database using the Data Handler microservice.

The collection's most important properties are the name ('properties.LUG11DESIG'), the area ID ('properties.BGRI11'), the polygon coordinates ('geometry.coordinates') and finally the coordinates of the center of the polygon that are used to get the weather data from the external Weather API ('center'), see figure 39.

5.4.2 WeatherAPI

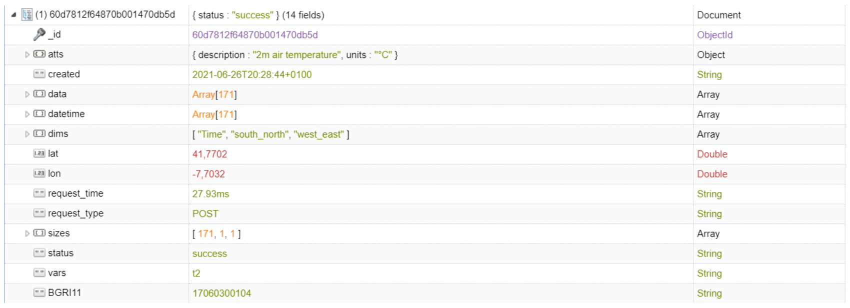
This collection contains data from the various weather variables available in the external API provided by the project colleagues, such as ambient temperature, relative humidity, wind speed.

The API receives the coordinates referring to the center of the polygon (area) of the map and the weather variable and returns the forecast for the weather variable in question for the next 7 days. The structure of the collection is represented in figure 40.



(2) 60c0c0decad57f0013230cbd	{ type : "Feature" } (5 fields)	Document
_id	60c0c0decad57f0013230cbd	ObjectId
type	Feature	String
properties	{9 fields}	Object
DTMN11	1706	String
FR11	28	String
SEC11	001	String
SS11	11	String
BGRI11	17062800111	String
LUG11	014068	String
LUG11DESIG	Vilar de Perdizes	String
Freguesias	170628	String
IV	12	Int32
geometry	{ type : "Polygon" } (2 fields)	Geo Polygon
center	{ type : "Point", coordinates : [-7.6304358925552345, 41.85530167520772] }	Geo Point

Fig. 39. Zones Collection



(1) 60d7812f64870b001470db5d	{ status : "success" } (14 fields)	Document
_id	60d7812f64870b001470db5d	ObjectId
atts	{ description : "2m air temperature", units : "°C" }	Object
created	2021-06-26T20:28:44+0100	String
data	Array[171]	Array
datetime	Array[171]	Array
dims	["Time", "south_north", "west_east"]	Array
lat	41,7702	Double
lon	-7,7032	Double
request_time	27.93ms	String
request_type	POST	String
sizes	[171, 1, 1]	Array
status	success	String
vars	t2	String
BGRI11	17060300104	String

Fig. 40. Weather Collection

5.5 Communication Between Components

In the previous sections, we have described each microservice, however, to fully understand the implementation of the proposed solution we will present how the different components communicate among each other to provide the desired services.

The communication between all components can be visible in figure 25.

5.5.1 *Dashboard and API*

The dashboard is the primary service that the user interacts with. It communicates with the API to obtain the necessary data to present to the user.

When the user enters the dashboard, the necessary data is requested to the API. The API checks if all the input requests are valid, and it communicates with the database to obtain the data. Finally, after the API has the data, it is returned to the dashboard to be presented to the user (figure 41).

The process encompasses three different requests: (1) obtain the map areas coordinates and the air temperature to each area; (2) relative humidity, and finally (3) the wind speed. The last two are only performed when the user clicks on the tab corresponding to each of the graphics in the detail view.

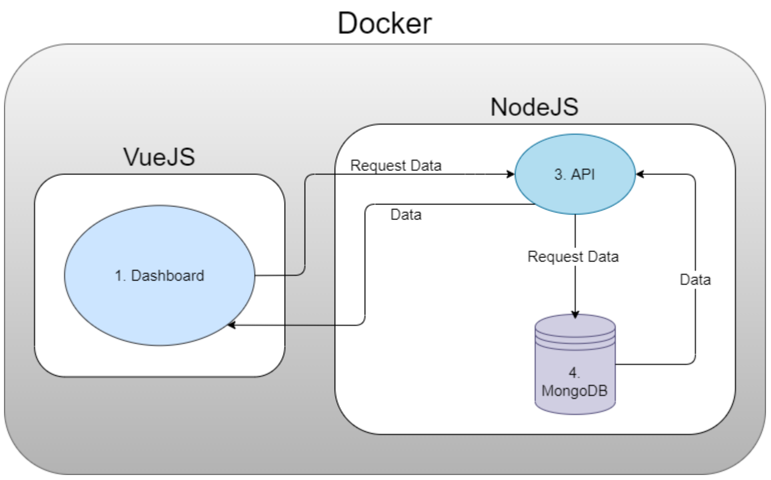


Fig. 41. Communication Between Dashboard and API

5.5.2 *Data Sources Handler and API*

Like we already mentioned, the Data Sources Handler microservice is used to upload 'static' data sources like, for example, a JSON or Excel file, so this communication occurs when an admin user wants to add data from new or existing data sources in the database.

The admin uploads the file with the data into the Data Sources Handler microservice, and, after this, the data is sent by an HTTP request to the API, which will add it into the database (figure 42).

If the data is added to the existing data source, the data would be available to consume in the application without having to make any changes.

On the other hand, if we add a new data source, we have two possibilities: (1) We can use the global route implemented in the API to get the data of any collection passing the name, or (2) we can implement the specific logic for the new data source in a separate module of the API.

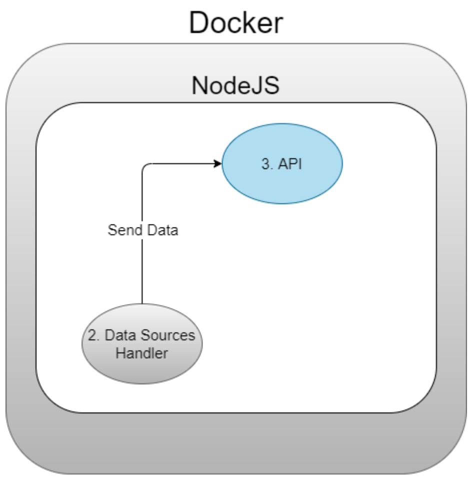


Fig. 42. Communication Between Data Sources Handler and API

5.5.3 API and Database

It is the most obvious communication of the application. To have access to the data, whether, for writing or reading purposes, the API must communicate with the database. As we already refer to before, this communication occurs through HTTP requests when a user enters the dashboard to obtain the necessary data, when an admin needs to insert new data from a file or when the job gets weather data from the external API (figure 43).

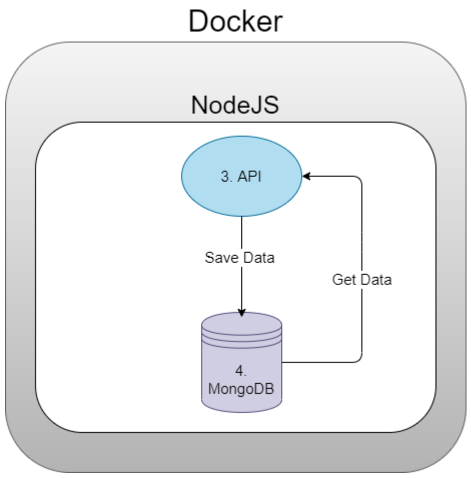


Fig. 43. Communication Between API and Database

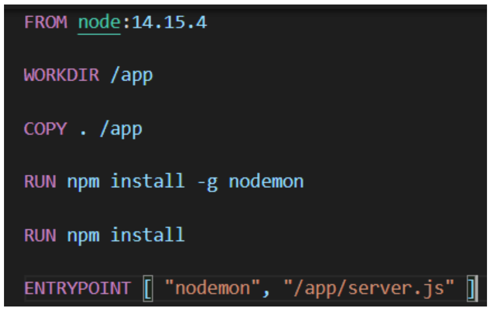
5.6 Setup and Configurations

For the application to work without problems in any location, it is necessary to carry out basic configurations. This section presents the necessary configurations to deploy a new installation of the proposed system.

5.6.1 Docker

For all dependencies to be installed automatically, it is advisable to use docker to compile and run each microservice in a container.

For this to be possible, each microservice has a Dockerfile with all the dependencies and configuration needed to generate a Docker Image. We can run the command "docker build ./" in the directory where the Dockerfile is located to create the image. In the figure 44 we can see an example of a Dockerfile.

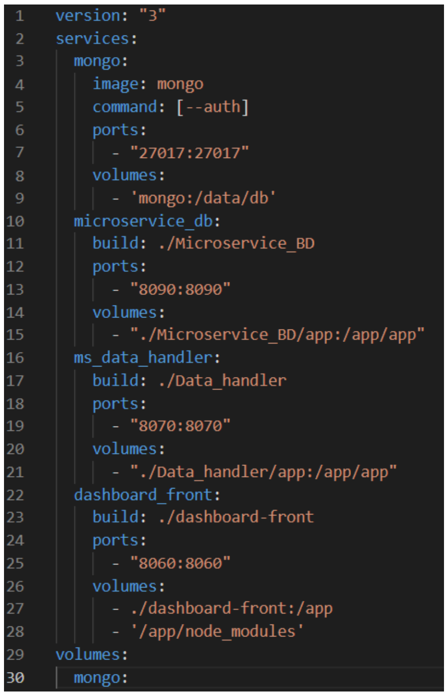


```
FROM node:14.15.4

WORKDIR /app

COPY . /app

RUN npm install -g nodemon

RUN npm install

ENTRYPOINT [ "nodemon", "/app/server.js" ]
```

Fig. 44. Dockerfile Example

After we have the images compiled, to run the Docker containers based on the images created before, we need to run the command "docker run { image name }" for each image.

If we want to compile all the images and run all containers simultaneously, we can use the Docker Compose. It is a more complex file, but by running the following command "docker-compose up –build" we compile all images and run all containers based on these images which starts a fresh instance of the dashboard (figure 45).



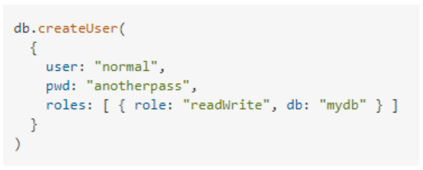
```
version: "3"
services:
  mongo:
    image: mongo
    command: [--auth]
    ports:
      - "27017:27017"
    volumes:
      - 'mongo:/data/db'
  microservice_db:
    build: ./Microservice_BD
    ports:
      - "8090:8090"
    volumes:
      - "./Microservice_BD/app:/app/app"
  ms_data_handler:
    build: ./Data_handler
    ports:
      - "8070:8070"
    volumes:
      - "./Data_handler/app:/app/app"
  dashboard_front:
    build: ./dashboard-front
    ports:
      - "8060:8060"
    volumes:
      - ./dashboard-front:/app
      - '/app/node_modules'
volumes:
  mongo:
```

Fig. 45. Docker Compose File

5.6.2 Database User Configuration

To have API access to the MongoDB database, we need a microservice running a Mongo instance with a configured user with read and write access to the project database.

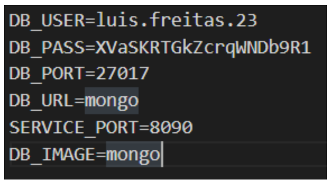
To create this user, we need to enter into the MongoDB container and execute the command in figure 46. We need to specify the username, password, and roles we want the user to have for the specified database name.



```
db.createUser(
  {
    user: "normal",
    pwd: "anotherpass",
    roles: [ { role: "readWrite", db: "mydb" } ]
  }
)
```

Fig. 46. Add Database User

Finally, after we have the user created, the username, password, and the port of the database container needs to be added in the API service configuration file (.env),



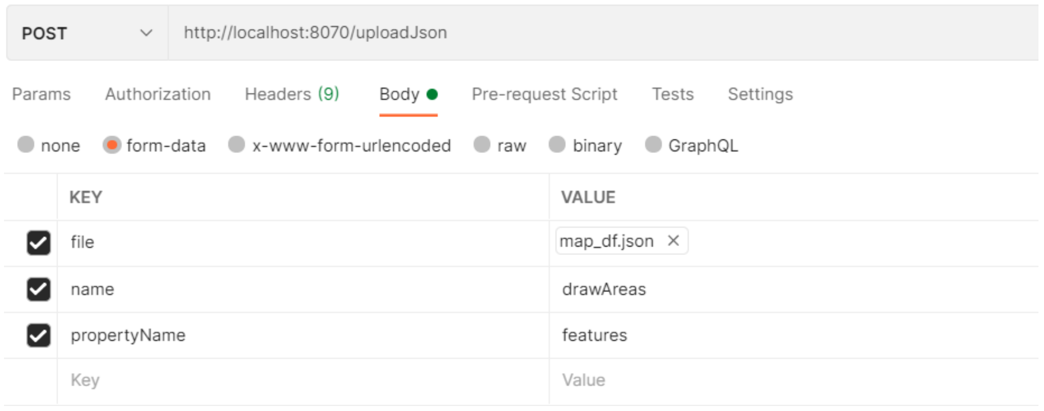
```
DB_USER=luis.freitas.23
DB_PASS=XVaSKRTGkZcrqWNDb9R1
DB_PORT=27017
DB_URL=mongo
SERVICE_PORT=8090
DB_IMAGE=mongo
```

Fig. 47. API Configuration File

5.6.3 Add Data to the Database

As mentioned above, the information about map area coordinates comes from a 'static' JSON file that RELIABLE project managers have provided. This data was added to the database.

Figure 48 shows an example of entering these data into the database. We need to specify the database collection name ('drawAreas') and the JSON property that has the data we need, in this case ('features').



http://localhost:8070/uploadJson

Body: [✓ form-data]

	KEY	VALUE
✓	file	map_df.json
✓	name	drawAreas
✓	propertyName	features
	Key	Value

Fig. 48. Add Map Areas Data

6 Evaluation

Motivated by the fact that at this moment in Portugal there is no system with the capabilities of the system proposed in this thesis, and the extreme atmospheric conditions have a great impact on people's mortality due to the poor thermal conditions of the buildings.

The objective of this thesis was to create an atmospheric conditions dashboard that consumes a data model that relates a set of variables such as weather data, thermal building conditions, number of daily deaths, and creates as output a risk index.

It has to be flexible for new data sources and new parameters within existing data sources and expandable so that it can be expanded to the entire national territory in the future. It has to have an excellent spatial resolution when compared to other existing systems.

The interface needs to be understandable and straightforward to be easily used by the general public. The dashboard API will be available to be consumed by other applications that want to use this type of data.

In order to verify that these objectives have all been achieved, we carried out the following evaluation methods:

6.1 Usability Tests

To verify if the final result of the dashboard meets the initial usability requirements, that is, the interface is responsive, has a good organization of information, and can be easily used and understood by an inexperienced user, we run usability tests with real users.

The usability of the interface was assessed through the think-aloud protocol. Think-aloud is among the most popular Human Computer Interaction techniques for usability evaluation [11]. Originally proposed by Ericsson and Simon [41], the method consists in asking participants to verbalize their thoughts as they use an interactive system to perform a set of pre-defined tasks. The big advantage of using think-aloud is that it does not only help identify usability issues but also gain insights into the overall user experience [59].

For the purpose of this study, the traditional think-aloud protocol was used [41] – i.e., the test moderator has minimal or no interactions with users and does not provide them with any kind of assistance. In order to observe how potential users would interact with the dashboard and identify major misconceptions, the following five tasks were given to participants:

(1) Find on the map a specific address (provided by the moderator);
(2) Identify the ID number of the area that the address belongs to;
(3) Find the current risk level of a specific area (ID of the area to find was provided by the moderator);
(4) Find the current speed wind in that area;
(5) Identify the current areas with higher and lower risk, respectively.

Five people participated in the test – a number of subjects that, as suggested by Nielsen [62], are sufficient to identify the majority of the issues with the system. The study started with a brief introduction to the think-aloud protocol, followed by a general description of the dashboard and its purpose. The sessions were carried out individually and recorded with permission from the participants. Each session lasted around 15 minutes. During the test, a moderator silently observed the participant performing the tasks and noted relevant events. At the end of each section, follow-up questions were asked for clarification if needed. All participants (three females and two males) are researchers at the Interactive Technology Institute. Their age ranges from 32 and 36 years (average age was 34,2 years).

The results were obtained by analyzing (i) task performance and (ii) verbalization data. Specifically, task performance was determined considering whether the task was completed correctly or not, and the degree of difficulty experienced before successfully completing a task:

- good performance = 0 or 1 wrong clicks;
- moderate = between 2 and 3 wrong clicks;
- poor = $\geq$ 4 wrong clicks or not completed.

Participants' verbalizations were transcribed and thematically analyzed using a general inductive approach.

6.2 Portability Tests

Another important point is the application portability. The entire application must be portable, that is, it must be easily installed and configured on any device. As such, we will be evaluating the portability of our solution. For this, we will install and configure the application on another device, which will later use to carry out usability tests with users.

7 Results

In addition to all the work carried out in the literature review and in the development of the application itself, it is essential to conduct tests on the work carried out to verify whether it achieved the intended objectives.

In this section, we will talk about two types of tests performed, namely portability and usability tests.

7.1 Portability Tests

User testing is a fundamental step in the application development process. For these tests to be possible, with some users, and because we cannot have access to a remote server, we installed the application from scratch on another Windows desktop.

It is essential that the application is easy to install on any platform, hence we have used Docker technology.

The installation process is very similar to the one presented in section 5.6, but here we will introduce some more details and draw some conclusions from the installation process.

This process is divided in seven steps:

7.1.1 Docker Desktop Installation

Docker installation is fundamental to compile and run our application without having to worry about dependencies. As the computer where the application was installed has a Windows 10 system, we can install Docker more simply by installing the Docker Desktop application.

The download and installation process is straightforward. Just follow all the steps presented. After the application is installed it looks like this (figure 49).

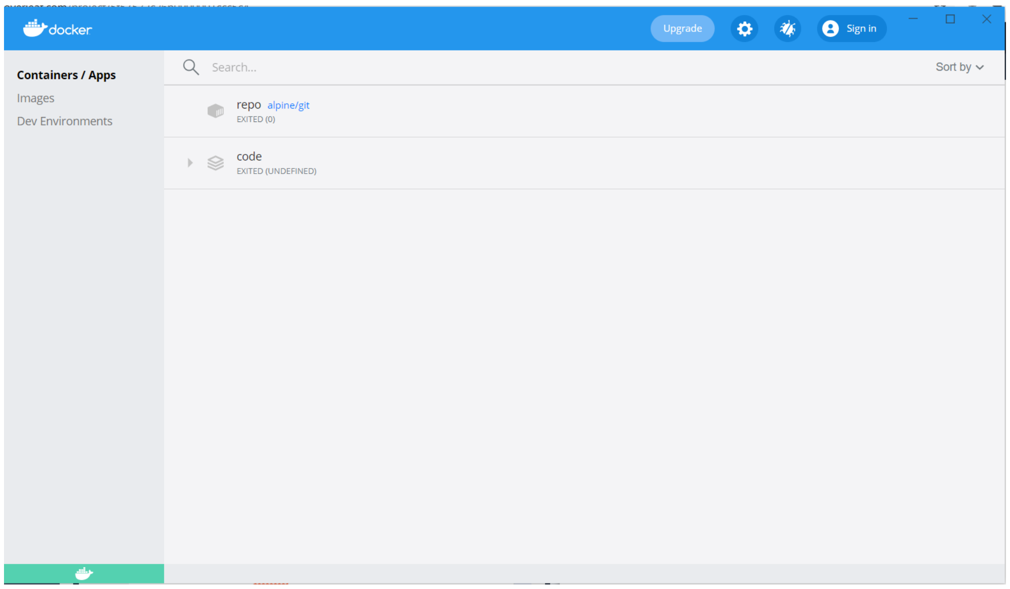
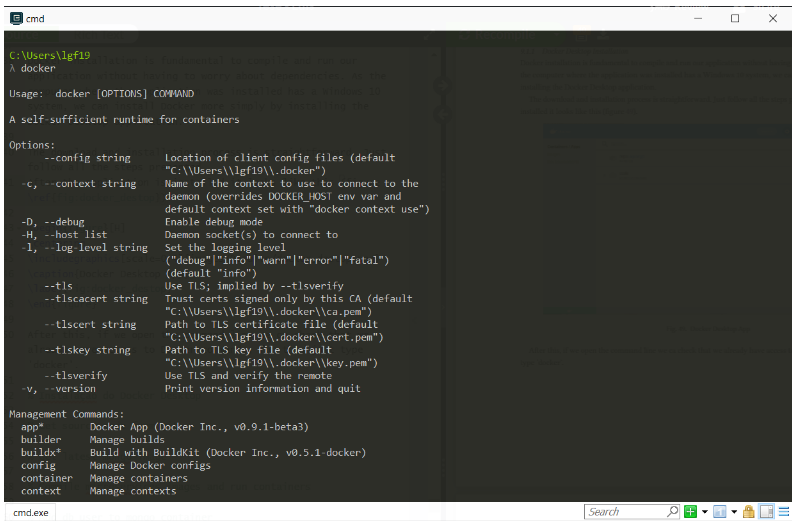


Fig. 49. Docker Desktop App

After this, if we open the command line we can check that we already have access to Docker commands, just type 'docker' like in next figure 50.



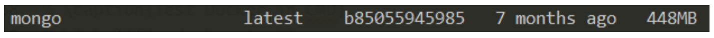
```
C:\Users\lgf19
λ docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
      --config string      Location of client config files (default
                           "C:\\Users\\lgf19\\.docker")
  -c, --context string     Name of the context to use to connect to the
                           daemon (overrides DOCKER_HOST env var and
                           default context set with "docker context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket(s) to connect to
  -l, --log-level string   Set the logging level
                           ("debug"|"info"|"warn"|"error"|"fatal")
                           (default "info")
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default
                           "C:\\Users\\lgf19\\.docker\\ca.pem")
      --tlscert string     Path to TLS certificate file (default
                           "C:\\Users\\lgf19\\.docker\\cert.pem")
      --tlskey string      Path to TLS key file (default
                           "C:\\Users\\lgf19\\.docker\\key.pem")
      --tlsverify          Use TLS and verify the remote
  -v, --version            Print version information and quit

Management Commands:
  app*        Docker App (Docker Inc., v0.9.1-beta3)
  builder     Manage builds
  buildx*     Build with BuildKit (Docker Inc., v0.5.1-docker)
  config      Manage Docker configs
  container   Manage containers
  context     Manage contexts
```

Fig. 50. Test Docker in CMD

7.1.2 Obtain Mongo Image

To have our database available, we need to have a Mongo image running within our Docker environment. Docker already provides a ready-made image of Mongo, and we have to get it, running the following command, 'docker pull mongo'.

After the download process is complete, if we run the command 'docker images', we can verify that we have an image named 'mongo', as shown below (figure 51).



```
mongo                  latest     b85055945985   7 months ago   448MB
```

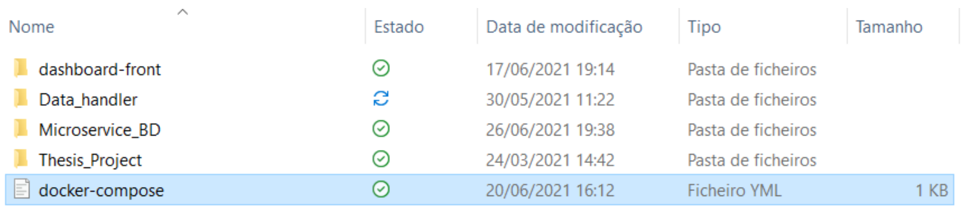
Fig. 51. Mongo Image

7.1.3 Get Application Source Code

We already have one of the necessary images compiled (mongo's) however, we need to have the images of the other microservices of the application. It was necessary to access their source code, which is available on two sites, GitHub and OneDrive.

In this case we just download the code from the OneDrive. After the download is finished we already have the source code on our machine.

As we can see, in the project folder, we have a file called 'docker-compose' (figure 52). This file is responsible for compiling the images of each microservice as well as running the containers.



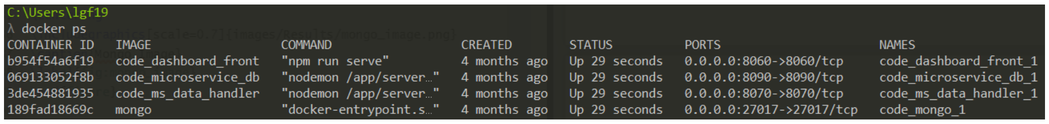
Nome	Estado	Data de modificação	Tipo	Tamanho
dashboard-front	✓	17/06/2021 19:14	Pasta de ficheiros	
Data_handler	⟳	30/05/2021 11:22	Pasta de ficheiros	
Microservice_BD	✓	26/06/2021 19:38	Pasta de ficheiros	
Thesis_Project	✓	24/03/2021 14:42	Pasta de ficheiros	
docker-compose	✓	20/06/2021 16:12	Ficheiro YML	1 KB

Fig. 52. Docker Compose File

7.1.4 Compile and Run all Microservices

After that, we only have to open the command line and go to the directory where the 'docker-compose' file is located. And finally run the command 'docker-compose up –build' to build the images and run the containers based on these images.

When processing is finished, we can verify that we already have our containers running, using the command 'docker ps' like in the next figure 53.



```
C:\Users\lgf19
λ docker ps
CONTAINER ID   IMAGE                   COMMAND                  CREATED        STATUS          PORTS                      NAMES
b954f54a6f19   code_dashboard_front    "npm run serve"          4 months ago   Up 29 seconds   0.0.0.0:8060->8060/tcp     code_dashboard_front_1
069133052f8b   code_microservice_db    "nodemon /app/server…"   4 months ago   Up 29 seconds   0.0.0.0:8090->8090/tcp     code_microservice_db_1
3de454881935   code_ms_data_handler    "nodemon /app/server…"   4 months ago   Up 29 seconds   0.0.0.0:8070->8070/tcp     code_ms_data_handler_1
189fad18669c   mongo                   "docker-entrypoint.s…"   4 months ago   Up 29 seconds   0.0.0.0:27017->27017/tcp   code_mongo_1
```

Fig. 53. App Containers

7.1.5 *Configure Database Access*

At this stage the application is already running. However, if we want access to the database, we have to create a database user with authorization to do so.

For this, it is necessary to have access to the container that contains the database. To facilitate the process, we are going to use the Docker Desktop application. After opening the application, we open the command line of the 'mongo' container, as shown in figure 54.

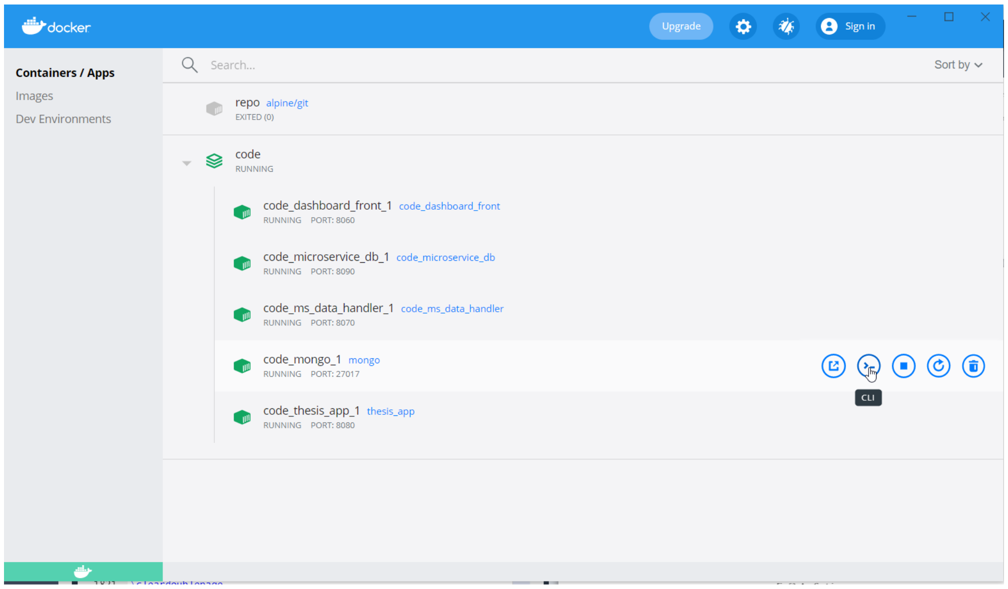
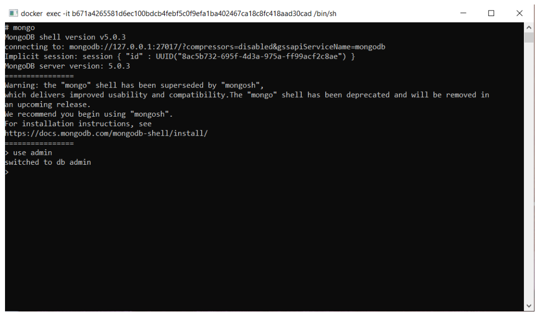


Fig. 54. Mongo Container Command Line

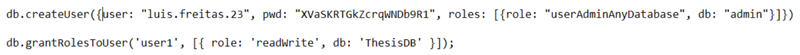
After that, inside the mongo container command line, run the following two commands ('mongo' and 'use admin') to get inside admin database (figure 55).


docker exec -it b671a4265581d6ec100bdcb4febf5c0f9efa1ba402467ca18c8fc418aad30cad /bin/sh

```
# mongo
MongoDB shell version v5.0.3
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("8ac5b732-695f-4d3a-975a-ff99acf2c8ae") }
MongoDB server version: 5.0.3
================
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and will be removed in
an upcoming release.
We recommend you begin using "mongosh".
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
================
> use admin
switched to db admin
>
```

Fig. 55. Admin Database

Finally, just run the following two commands (figure 56) to create the user with access to the project database (ThesisDB). The password and user names can be defined during the configurations, but if changed, they must be updated in the .env file from the API microservice.



```
db.createUser({user: "luis.freitas.23", pwd: "XVaSKRTGkZcrqWNDb9R1", roles: [{role: "userAdminAnyDatabase", db: "admin"}]})

db.grantRolesToUser('user1', [{ role: 'readWrite', db: 'ThesisDB' }]);
```

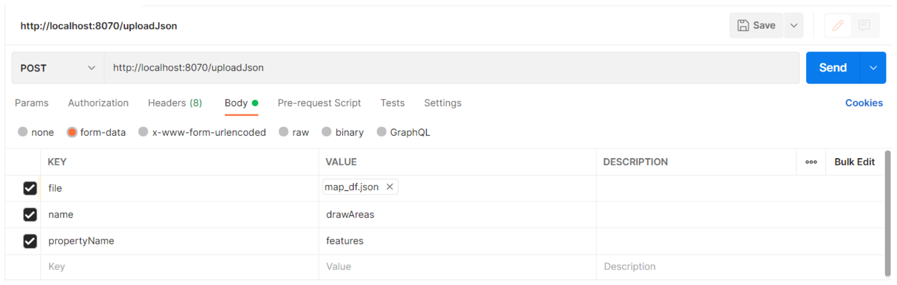
Fig. 56. Database User Access

7.1.6 *Add Initial Data*

Now that everything is ready to use the application, we have to add the initial data to work correctly. For this, we have to add data to the two essential collections for application, the data referring to geographic areas (drawAreas) and the meteorological data (weatherData).

We have prepared routes in the API for entering this data. We only have to make one HTTP request for each type of data, and the data will be added to our database. To facilitate the process, we are going to use the Postman application.

Figure 57 shows the request to enter the geographic data of the areas to be marked on the map.



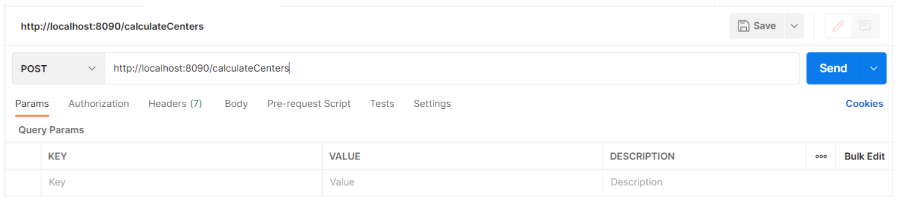
http://localhost:8070/uploadJson

POST http://localhost:8070/uploadJson

	KEY	VALUE	DESCRIPTION
✓	file	map_df.json	
✓	name	drawAreas	
✓	propertyName	features	
	Key	Value	Description

Fig. 57. Upload DrawAreas Collection Data

After that, we need to run the job to calculate each area center point running the following HTTP request (figure 58).



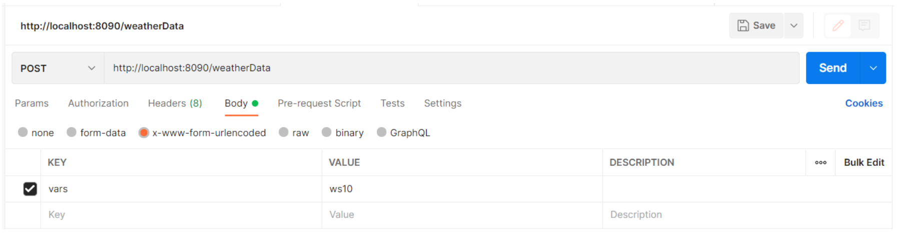
http://localhost:8090/calculateCenters

POST http://localhost:8090/calculateCenters

	KEY	VALUE	DESCRIPTION
	Key	Value	Description

Fig. 58. Calculate Centroid Point

Finally, just get the weather data for each central point in each area (figure 59). At this moment we are using three types of variables wind speed (ws10) and relative humidity (rh2) and air temperature (t2).



http://localhost:8090/weatherData

POST http://localhost:8090/weatherData

	KEY	VALUE	DESCRIPTION
✓	vars	ws10	
	Key	Value	Description

Fig. 59. Obtain Weather Data

7.1.7 Run the Application

Finally, we can access to our application. By accessing the URL configured for the frontend microservice, which in our case is http://localhost:8060, the result is the same like in figure 26.

7.1.8 *Summary*

In short, the installation of the application was done from scratch. Taking this into account, we can say that it was a simple and relatively quick process.

We did not have to worry about the dependencies needed for each of the microservices. This was because we are using docker technology, as its objective is to create this abstraction layer so that the installation processes are independent of the machine we are installing.

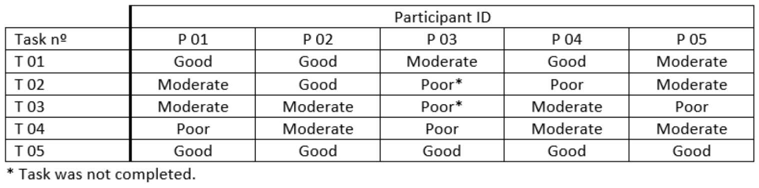
Therefore, we can say that the goal of application portability was successfully achieved.

7.2 Usability Tests

Here we will present the results of the execution of the proposed tasks to the users during the tests. First, let's talk about showing the result of the execution of the five tasks. Then on the verbalization data and finally present some application limitations that emerged from the tests.

7.2.1 *Task Performance*

In terms of task performance (see table below 60), three out of five tasks were successfully completed by all participants, while one participant did not complete tasks two and three.



	Participant ID				
Task nº	P 01	P 02	P 03	P 04	P 05
T 01	Good	Good	Moderate	Good	Moderate
T 02	Moderate	Good	Poor*	Poor	Moderate
T 03	Moderate	Moderate	Poor*	Moderate	Poor
T 04	Poor	Moderate	Poor	Moderate	Moderate
T 05	Good	Good	Good	Good	Good

* Task was not completed.

Fig. 60. Task Performance Results

Task 01: Find on the map a specific address (provided by the moderator): all participants completed the task successfully. Only two out of five encountered minor issues. Specifically, it was pointed out that when typing the address provided by the moderator in the map search box, the autocomplete search box does not return any suggestions when adding the street number. This led one participant (P 03) to think (s)he inputted the wrong address. Additionally, both participants (P 03 and P 05) reported that the search box is too short (see figure 61); thus it does not allow to read long addresses entirely. Ultimately, three out of five participants were disappointed by the fact that the map search query does not result in a pin displayed on the map. Such feedback was described as extremely important to understand whether the map search was successful or not.

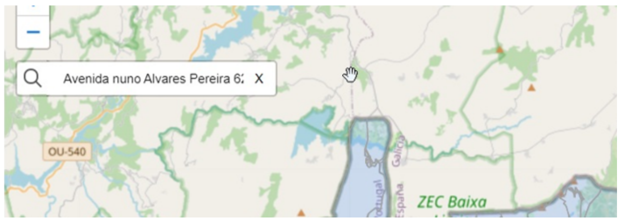


Fig. 61. Map Search Box

Task 02: Identify the ID number of the area that the address belongs to: only one participant completed the task without encountering any difficulties. All the others found this task hard to perform since they couldn't understand that the map is interactive – i.e., they can click on it to open a tooltip with additional information related to the area selected. In this regard, it was suggested to add affordances – e.g., the tooltip could automatically open as a result of the map search.

Task 03: Find the current risk level of a specific area (ID of the area to find was provided by the moderator): all participants experienced some difficulties in performing this task, which all relate to the identifiers used to refer to the subsections – i.e., a sequence of eleven numbers that is not 'meaningful' to users. In this regard, P 03 said, "I didn't notice them. I saw numbers and thought it was just another piece of data". When told that those are the identifiers officially used by the national authorities to classify geographic information and that the subsections do not have a 'name', P 03 suggested to help users link the identifiers with the actual geographic locations they refer to by linking map and sidebar – i.e., once an area is selected on the map, the respective identifier is highlighted in the sidebar. Another user suggested grouping the subsections by parish so as to facilitate the search. Another aspect worth mentioning is that two participants (P 03 and P 05) tried to search for the subsection by using the search box on the map, suggesting it could be relevant to include this feature in the system.

Task 04: Find the current speed wind in that area: regarding this task, all the issues experienced by participants are due to the tab menu used to organize the different pieces of data (i.e., risk level, humidity, and wind speed). In fact, the arrow used to navigate within the tab menu items only allows moving to the previous/next tab; therefore, it does not work when the next/previous tab is visible (see figure 62). In other words, in order to reach the last tab (i.e., wind speed), users needed to select first the second tab (i.e., humidity) and only then use the arrow to access the following tab. This bottleneck can be easily overcome by modifying the way users navigate the tab menu.

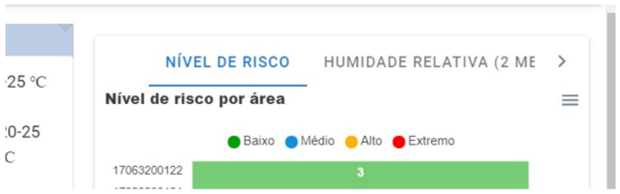


Fig. 62. The Tab Menu

Task 05: Identify the current areas with higher and lower risk, respectively: all participants successfully completed the task without any difficulty.

7.2.2 Verbalization Data

The following four main themes emerged from the analysis of the verbalization data:

Colors code: three out of five participants pointed out that the color codes used to classify the different data sets (e.g., temperature and risk level) were reported to be too similar and consequently a source of confusion. When performing task 03, one participant started questioning whether the color of the bar represents the temperature of the area or the risk level "the risk should be high since the bar is light yellow. Or maybe that's the temperature, and 30 is the level of risk. I mean, it seems to be the same color as one of those used in the legend on the map. Mmmm…no…actually, it's just similar. And the table has its own legend…I was wrong. The risk is 'high'".

Interactive features: several participants reported having issues understanding what elements of the interface are interactive and what are not. In particular, the main source of confusion was the map. As mentioned above, participants suggested adding some prompts to the map – e.g., a tooltip (opening automatically or on mouseover) that provides some information about the area and suggest clicking on it to know more.

The different views should be linked together: another aspect that emerged from the think-aloud exercise is the lack of a link between the information displayed on the map and the data provided in the sidebar. While performing task 05, a participant mentioned that (s)he would have liked to see where those sub-sections were located: "I wonder if these areas are distant from each other or not. It would be useful if, after selecting an area from the chart, the same area gets highlighted on the map". Also, participants' verbalizations suggest that such a link between the elements of the interface could facilitate information retrieval. For example, P 02 mentioned: "it would be easier to find the risk level of an area simply by selecting it from the map".

Geographic information facilitate understanding: finally, several participants reported appreciating the fact that the map is the main element of the interface. P 03, in particular, stated: "I like that the first thing that catches my attention is the map. Usually, data are provided in the form of numbers while the geographic information provides context to it". Moreover, the first comment that P 01 made when presented to the interface was: "a map! This will make it easier to look for information".

7.2.3 Limitations

Although not central for this Thesis project, the user study was conducted with the goal of extending the scope of the evaluation so as to include an initial assessment of the front-end. Despite some limitations, the feedback collected provided a set of useful insights that could be implemented as future work. In this regard, two are the main limitations of this study: the sample size and the characteristics of the participants. Although sufficient to identify the main usability issues of an interactive system, a bigger sample would have provided richer insights on the usability of the interface. Also, it should be pointed out that participants in the study are Human Computer Interaction researchers - therefore not the actual end-users - and not very familiar with the geographic area involved (the demo focuses on Montalegre). As part of the RELIABLE project, it was initially planned to conduct the study in Montalegre as part of a workshop with the local stakeholders. However, as a consequence of the pandemic, the workshop was canceled. It was therefore decided to opt for this specific sample, with the goal of getting feedback from people that are experienced in usability and the possible bottlenecks that specific design choices can engender.

8 Discussion

This section will review all the work presented before to conclude what went well and what could have gone better.

We will discuss the following points (1) if each Proposed Solution item resolves a problem presented in the Problem Statement, (2) if the technologies presented in the Literature Review were appropriate, (3) if the data sources were the most suitable, (4) and how we compare the implemented solution with the revised work. This discussion will also include other observations gathered during the development of this thesis.

8.1 Assessing Extreme Events

The Problem Statement section focuses on two existing Portuguese solutions, the ICARO and FRIESA systems, that evaluate the risk level of extreme heat and cold weather events, respectively. As we do not have direct access to these two systems, an analysis was carried out based on information provided by the RELIABLE project partners. The objective of this analysis was to find the weaknesses of these systems so that we could implement a solution that would solve them.

These systems are not available for the general public to consult. Only competent entities such as health institutions can access them, and there is no single central point to access weather data for heat and cold. We addressed this problem by developing a Web Application for heat and cold events open to any citizen with an internet connection. This access is not yet possible because the application has not yet gone into production, but it will be available when that happens.

Another disadvantage is that the risk assessment carried out by these systems does not cover the entire Portuguese territory. For example, the FRIESA system only works for the cities of Lisbon and Porto. We believe that all places in the country must have access to this information. With this objective in mind, and as we still do not have access to data for the other country areas, the application was implemented for the Montalegre area as a pilot test. But in a way that it could be easy to expand to the rest of the Portuguese territory in the future, it will only be necessary to introduce new data. That data will be available in the dashboard to be consulted by its users. In this case, it would only be required to add information about the new zones in the 'DrawAreas' collection. The structure of this information is shown in figure 39. The rest of the process of obtaining meteorological data for each zone will work in the same way because it only needs the areas' coordinates.

The systems in question have a low spatial resolution. The weather forecast is for large geographic areas like an entire municipality, and the forecast will be the same for the whole area. This aspect makes the forecast less reliable, as if we imagine a municipality can have a large territorial area. As we know, the temperature or risk is different depending on the zone. Our implemented solution uses high spatial resolution: weather forecasts, risk index calculation, and all other estimates are made for a smaller geographic area that we call subsections. This way, each specific area can have a more accurate forecast.

Finally, we should consider another aspect in a more reliable and flexible system for future changes. Both systems are based on the weather forecast and the number of deaths to predict the risks of temperature (high or low) in mortality in some regions of the country. However, temperature is not the only important variable for predicting mortality during extreme weather events. Other variables, such as the thermal conditions of buildings, the population's financial possibilities, and age, greatly influence the way people manage to protect themselves. The REALIABLE project took these aspects into account and created a model that considers all of them to make a reliable forecast of the level of risk for each area while being flexible to add new variables in the future. This model was not finished in time by the rest of the RELIABLE project team. However, everything is prepared so that it can be integrated without any problems.

8.2 Solution and Implementation

At the architectural level, and based on our two primary goals for this project, scalability and flexibility, a survey was carried out to determine which architectural pattern would best suit our purposes. Considering the advantages and disadvantages of the different architectural patterns, we have decided to follow a microservices architecture. Despite being recent, it has been widely used in applications intended to be as flexible and scalable as possible. For example, if we need to add another service or functionality to our application, we can create a new microservice without changing the existing ones.

Finally, we associate Docker technology to this architecture so that the application becomes independent of the system where it will run, the necessary dependencies are installed automatically. This way, we can easily install our solution on any system.

At the level of technologies used for implementation, a survey was carried out during the literature review phase on which technologies could help implement our solution. The project's base framework remained the same as the one we had chosen in the literature review, NodeJS, as it is widely used in microservices architectures as well as helping create flexible and scalable applications. This technology was applied to all the application's microservices, except for the one that contains the database, because it does not have any business logic, it only contains the database itself. This option brought us architectural and structural advantages in terms of a vast set of already implemented libraries that we only have to install and use. For example, the 'vue2-leaflet' dependency links OpenStreetMap and Leaflet and gives us a set of components that we can use directly in Vue. This particular dependency makes our work much easier because it already offers us a set of abstractions that made the whole part of drawing the map, showing data on the map much simpler.

On the front-end, in addition to Nodejs, we added the VueJs library to fulfill yet another objective of our solution based on the simplicity of use and responsiveness of the same in different equipment such as a desktop, tablet, or mobile phone. It made our work straightforward again, as this library already offers a set of responsive components without creating the CSS of the elements ourselves from scratch.

At the level of the detail view, we needed to show a set of data in more detail, using graphs. During the Literature Review, we made the first choice to use the Chart.Js library to treat charts. Still, later on, when we started trying to develop this part, we chose to use another library, APEXCHARTS, since it is much simpler to integrate with Vue.js.

Finally, we chose to use MongoDB technology to meet the application's flexibility and expandability requirements at the database engine level. There are no dependency issues between database tables, and data processing is universal, as everything is in JSON format. If we want to add a new data source, we create a new collection, and if we are going to add a new variable to an existing data source, we have to add this unique property to the collection, and everything else remains the same.

In short, in our opinion, the technologies chosen were the most correct as they allowed us to create a flexible and scalable application that fulfills the project goals. In terms of design, creating a single-page dashboard with well-organized and structured information also allowed us to achieve one of the goals that involved the simplicity of the application so that any user, even inexperienced, can use our application without any problem.

In terms of portability, as we can experience in section 7.1, the installation and configuration of the application is simple. This is because the Docker technology is used. It creates a whole virtualization environment that makes us not have to worry about the dependencies to be installed, machine resources, and others.

8.3 Data sources

In an initial phase of the RELIABLE project, when reviewing the literature of this thesis, we had as data sources the data from the 2011 censuses, the ICARO and FRIESA systems, and finally, the database with the energy certificates of the buildings. Later, after more developments, the data sources used to carry out this thesis were different from the initial ones (will be explained at the end of this section). Currently, three data sources are consumed from the application.

The first and one of the most important is the source that provides us with the information about the necessary geographic zones to be drawn on the map, such as geographic coordinates, name, and identifier. These areas are currently restricted to the district of Montalegre and were provided in an Excel file whose content was stored in the project's database using the Data Sources Handler microservice.

There is also an API that gives us a weather forecast for the next seven days. Combining the previous data source with this service, we can obtain information on several meteorological variables such as air temperature, wind speed, relative humidity, etc. This information is used in the map popup to show the air temperature of each zone and in the graphs of the detail, view to show data from other meteorological variables such as wind speed and humidity.

Finally, as the third and final data source, we have a model that calculates the risk level for each zone. This is where the initial project data sources are referenced, as this model uses all of them to estimate a risk level. Variables such as atmospheric conditions, thermal conditions of buildings, and population age are used to generate a more reliable risk index. As already mentioned, this has not yet been finalized by the RELIABLE project team, and we are currently using the air temperature as a risk indicator. However, everything is ready to be easily replaced. It is probably the most important data source as it will allow us, based on all the others, to estimate the level of risk for each zone so that people can take preventive measures.

8.4 Contributions to the field

For the successful development of our application, the review of the existing state of the art was also critical. In this case, in the literature review, we reviewed applications related to extreme atmospheric events and possible pandemics such as COVID-19. The review of these applications allowed us to perceive what already exists and how it is built, at least visually. With this review, we can conclude what we could use, and improve, in our application.

We chose to create a single-page dashboard since all the reviewed work opted for this option, except the IPMA portal. In this way, the information is more centralized and accessible, the user does not need to navigate from page to page, thus preventing the user from forgetting the data he saw on the previous page.

Still referring to the interface and the content organization, we tried to be straightforward as possible so that non-experienced users can easily and quickly consult the needed information. For the layout, we were based on the ViewExposed application, which is divided into three parts. The left half of the screen is occupied by the map as we think it is the most crucial area and where users will quickly find more information. There is also a detailed area in the right upper corner, with the graphics showing more specific information about map polygons. Finally, the lower-left corner presents helpful information and contacts.

Another point we wanted to improve was to have a better spatial resolution than existing applications. For the IPMA website, the minimum spatial resolution that allows visualizing information is city/town level. If we think about it, it is still a considerable space. Since meteorological data is not the same for all points in an area with these dimensions, the forecast presented might not be correct. We decided to implement a higher spatial resolution so that the data presented to the user is as reliable as possible. To improve the prediction of the level of risk, instead of only using the air temperature for the estimate, we decided to use other variables that also influence the imminent risk to adverse weather conditions, such as the thermal conditions of buildings, people age.

Finally, and to take advantage of the study made in a set of applications that gave rise to a group of good principles for creating a dashboard in section 2.1.8. We decided to use these same principles in our application since this way, we would be establishing our design on a solid base.

9 Conclusion

Climate change is increasingly impacting the Earth's surface, thus affecting everything on it. Human beings are one of the main species affected by several extreme events, such as heavy rain and extreme heat. There must be learning and a consequent change in human behavior to mitigate the consequences. Several governments are implementing a series of regulations so that global warming is minimized in the future. The consequences have a greater or lesser impact depending on a set of factors. For example, the thermal conditions of buildings are essential as they will allow people to feel comfortable inside their homes. The population's age and associated diseases are also a factor to consider, as older people or people with related diseases are not so resistant to this type of extreme events.

In Portugal, as mentioned above, we know that the population is aging and that the thermal conditions of the buildings are not the best due to their age and the low income of families. We also know that there is currently no open platform for the public to consult atmospheric conditions with a high spatial resolution. Keeping the population informed about possible extreme weather events is an excellent weapon in the fight against the consequences they can cause and the analysis of events in the past so that the prediction and preventive measures can be more effective in future events.

This thesis intended to do just that, that is, to offer people a meteorological dashboard with a higher spatial resolution than any of the current ones, the proposed system data refers to an area smaller than that of a parish. Another goal was that it would be easily accessible by any user without the need for user experience. In addition to weather data, the main output is the risk index displayed for each of the areas marked on the map. This index lists a wide range of variables, such as meteorological variables, thermal conditions of buildings, age of the population, among others. We are offering a new risk indicator that does not currently exist in our country.

As the last goal, the application had to be easily scalable and flexible enough to be expanded to the entire Portuguese territory and allow the addition of new data sources or variables. The application was built and designed using standards and technologies that allow us to do this. For example, when the risk index is finalized by those responsible for the RELIABLE project, everything is prepared on the application side so that this new index easily replaces the air temperature index.

The experience of developing the system proposed by this thesis was excellent, as I had never developed anything of this dimension previously. It was a complete process, from the investigation phase of the problem itself, from what is already implemented to minimize the consequences of climate change, to the step of requirements interpretation, architecture design, and implementation.

In addition to all this, the thesis is part of a research project, RELIABLE. It was an advantage as I had not yet participated in any project outside the autonomous region of Madeira, let alone with this dimension. It was an excellent experience to see the processes used in a project of this type. It was very gratifying to recognize the work developed in this thesis by those responsible for the project. It will be even more so when the platform goes online for the general public.

In terms of results, the usability tests were going to be carried out by project managers at a congress held in the district of Montalegre. However, due to the COVID-19 pandemic, the congress was been postponed and the user's tests could not be carried out in a valuable time for this thesis. To combat this lack, tests were carried out here in the region with some users. As usual, the tests resulted in less good and good aspects of the application. But in general, the application was well accepted by users, who found it an asset. The points presented as limitations in the results of usability tests will be improved as future work.

Regarding the integration tests, it was impossible to perform them at the end of the development of this thesis due to lack of time. However, throughout the development phase, new data sources emerged, and in all of them, it was possible to add them without conditioning the application's operation with the existing data sources. So, we can say that the application is well modularized.

Finally, portability tests, we can conclude that the use of the docker was an advantage to the application. This way, we could easily install the application anywhere without worrying about dependencies due to the docker's virtualization.

The RELIABLE project had other objectives, such as developing a notification system to alert users to potentially dangerous situations. However, it was not one of the main objectives of this thesis. It will be implemented in the future at the level of the project.

Briefly, we can argue that, the main objectives of the thesis were achieved. We have a complete dashboard with a large spatial resolution, which lists several variables (atmospheric variables, building conditions, population age, among others), is simple to use for non-experienced users, is open to the general public, and is flexible and scalable for the future changes.

9.1 Future Work

In this section, we will expose what can be done in the future to improve the work of this thesis. The majority of work has been done, but always exists quick adjustments to be made.

9.1.1 Store Data from Weather API

At this moment, to obtain and insert the data from Weather API in our database, we need to execute an HTTP request manually, using, for example, Postman software. The rest of the process is automatic, we just need to make the HTTP request to a route of our API and the system takes care of the rest.

To resolve this detail is necessary to add a cron job to execute this HTTP request when needed in the background. With this, this process will be all automatic. This has not yet been done as the system has yet to go into a production environment.

9.1.2 Risk Index

The risk index is the main output of the RELIABLE project model. Based on it, we can see the level of risk of a specific zone on the map. Like we refer to in the Implementation section, we are using the air temperature instead of the risk level to present the risk of each zone because this risk is not finished yet by project partners.

As an initial requirement, the entire project would have to be flexible, and this part is no exception, so it will be easy to substitute one variable for another in the future. It is necessary to replace the air temperature request by risk index, and after this, use this data in the front-end of the application.

9.1.3 Add New Weather Variables

One of the requirements was adding new data sources or adding new variables inside an existing data source in the application. If in the future this need arises can be easily satisfied. For example, if there is the need to add a new graph with another weather variable in the Detail View of the dashboard, we need to make a new HTTP request to obtain the corresponding data and add a new chart in the front-end to show this data, without affecting the rest of the system.

Or suppose we need to add a new data source from a JSON or Excel file. In that case, we already have all the logic required to insert this data in the database using the Data Sources Handler Microservice. We just need to add the specific logic of this new data source in an isolated module of our API.

9.1.4 Expand to all Portuguese Territory

As we referred at the beginning of this thesis, we will expand the dashboard from the Montalegre district to all Portuguese territory. It is a task that does not require a lot of effort because we already have a scalable and flexible architecture.

We just need access to the coordinates of all other areas of the Portuguese territory, store this data in 'drawAreas' database collection, and after this, the process to draw the zones in the map, obtain weather data for each zone will be the same as today.

9.1.5 Polygon Centroid

The air temperature showed in the popup of each zone is the temperature for the centroid of polygon. So, no matter what site of the polygon the user clicks that the temperature will be the same.

A possible improvement solution, since this services requires a set of coordinates, it would be necessary to obtain the coordinates where the user clicked. Then, we could check if there was already a record of weather data in the database. Otherwise, it would have to be requested from the weather API of the RELIABLE project and saved in our database to avoid making requests to the API and historic.

9.1.6 Login and Notification System

There were other two requirements of the RELIABLE project, but they were not the main objectives of this thesis hence they were not implemented. They were there, a login and notification system. These two are related. First, the user creates an account, and after he will receive notifications on his device about dangerous situations.

As we are using a microservice architecture, it makes the application more extensible and scalable. When implementing this feature, it is advisable to create two new microservices, one for the login and another for the notifications. For the user to access the Dashboard and view the currently displayed data, an account is not required. In other words, our current services will not be affected, as we are not modifying the work that is already done but adding new features separately.

The login system will be used to user save their personal information and the areas for which they want to receive alert notifications. Regarding the notification system, it will be a background process that will check if there is a danger for the areas marked by the users, if there is, it will send a notification to the user's mobile device.

9.1.7 Other Tests

In section 6 we present a set of different types of tests that we intended to carry out for our application. However, due to some factors, such as the delay in the availability of data sources, it was not possible to carry them out within the thesis.

As mentioned in section 7, there was only an opportunity to carry out two tests: usability and portability. However, integration, overload, and performance tests were lacking.

As future work, it would be interesting to carry out these tests to evaluate parameters such as application responsiveness, ability to adapt to new data sources, and others.

9.1.8 Limitations

As a result of usability tests, we obtained positive aspects of the application, but we also had limitations. We intend as future work to resolve these points to improve users' use of the application.

References

[1] Analytics Across the Enterprise: How IBM Realizes Business Value from Big ... - Brenda L. Dietrich, Emily C. Plachy, Maureen F. Norton - Google Livros.

[2] AR4 Climate Change 2007: The Physical Science Basis — IPCC.

[3] AR5 Climate Change 2013: The Physical Science Basis — IPCC.

[4] Big Data: A Revolution that Will Transform how We Live, Work, and Think - Viktor Mayer-Schönberger, Kenneth Cukier - Google Livros.

[5] Business Intelligence and Analytics: From Big Data to Big Impact on JSTOR.

[6] Declining Vulnerability to Temperature-related Mortality in London over the 20th Century | American Journal of Epidemiology | Oxford Academic.

[7] Driving Rig Performance through Real-Time Data Analysis, Benchmarking, Dashboards and Developed Key Performance Indicators - OnePetro.

[8] Espaço Urbano: A Cidade e a Questão Ambiental - Brasil Escola.

[9] EUROMOMO EuroMOMO Bulletin, Week 40, 2020.

[10] Excess Winter Deaths in Europe: a multi-country descriptive analysis | European Journal of Public Health | Oxford Academic.

[11] Getting access to what goes on in people's heads?

[12] http://www.nytimes.com/2012/02/12/sunday-review/big-datas-impac. page 5.

[13] Instituto Português do Mar e da Atmosfera.

[14] Leaflet — an open-source JavaScript library for interactive maps.

[15] Measuring e-government impact | Proceedings of the 6th international conference on Electronic commerce.

[16] Models and Managers: The Concept of a Decision Calculus | Management Science.

[17] O que é SQL? | Alura Cursos Online.

[18] Overview | Maps JavaScript API.

[19] Ponto de Situação Atual em Portugal - COVID-19.

[20] Portal do INE.

[21] The Society for Conservation Biology.

[22] The Under-Exploitation of Natural Resources: A Model with Overlapping Generations* - KEMP - 1979 - Economic Record - Wiley Online Library.

[23] USA Severe Weather Reports - Dashboard.

[24] What is MongoDB? Introduction, Architecture, Features & Example.

[25] Transforming Our World: The 2030 Agenda for Sustainable Development. In William Rosa, editor, *A New Era in Global Health*. Springer Publishing Company, New York, NY, June 2017.

[26] Chart.js, October 2020. Page Version ID: 981400000.

[27] CSS, November 2020. Page Version ID: 989497438.

[28] HTML, October 2020. Page Version ID: 986056950.

[29] Java (programming language), November 2020. Page Version ID: 990749799.

[30] G. Andrienko and N. Andrienko. Exploring spatial data with dominant attribute map and parallel coordinates. *Computers, Environment and Urban Systems*, 25(1):5–15, January 2001.

[31] Liliana Antunes, Susana Pereira Silva, Jorge Marques, Baltazar Nunes, and Sílvia Antunes. The effect of extreme cold temperatures on the risk of death in the two major Portuguese cities. *International journal of biometeorology*, 61(1):127–135, 2017. Publisher: Springer.

[32] Joan Ballester, Xavier Rodó, Jean-Marie Robine, and François Richard Herrmann. European seasonal mortality and influenza incidence due to winter temperature variability. *Nature Climate Change*, 6(10):927–930, October 2016. Number: 10 Publisher: Nature Publishing Group.

[33] Adrian Gerard Barnett. Temperature and Cardiovascular Deaths in the US Elderly: Changes over Time. *Epidemiology*, 18(3):369–372, 2007. Publisher: Lippincott Williams & Wilkins.

[34] Lynn Baskett, Cynthia Lerouge, and Monica Tremblay. Using the dashboard technology properly. *Health progress (Saint Louis, Mo.)*, 89:16–23, September 2008.

[35] Arno Behrens, Stefan Giljum, Jan Kovanda, and Samuel Niza. The material basis of the global economy: Worldwide patterns of natural resource extraction and their implications for sustainable resource use policies. *Ecological Economics*, 64(2):444–453, December 2007.

[36] Jonathan Bennett. *OpenStreetMap*. Packt Publishing Ltd, September 2010. Google-Books-ID: SZfqR-cPXApoC.

[37] Hsinchun Chen, Roger H. L. Chiang, and Veda C. Storey. Business Intelligence and Analytics: From Big Data to Big Impact. *MIS Quarterly*, 36(4):1165–1188, 2012. Publisher: Management Information Systems Research Center, University of Minnesota.

[38] Margarida Costa, Beatriz S. Silva, Francisco C. Real, and Helena M. Teixeira. Epidemiology and forensic aspects of carbon monoxide intoxication in Portugal: A three years' analysis. *Forensic Science International*, 299:1–5, June 2019.

[39] Keith B. G. Dear and Anthony J. McMichael. The health impacts of cold homes and fuel poverty. *BMJ*, 342, May 2011. Publisher: British Medical Journal Publishing Group Section: Editorial.

[40] Wayne W. Eckerson. *Performance Dashboards: Measuring, Monitoring, and Managing Your Business*. John Wiley & Sons, November 2010. Google-Books-ID: 5nuYDwAAQBAJ.

[41] K. Anders Ericsson and Herbert A. Simon. Protocol Analysis: Verbal Reports as Data. April 1993.

[42] Stephen Few. *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly Media, Inc., 2006.

[43] Paul H. Fischer, Bert Brunekreef, and Erik Lebret. Air pollution related deaths during the 2003 heat wave in the Netherlands. *Atmospheric Environment*, 38(8):1083–1085, March 2004.

[44] César Fonseca, Manuel Lopes, and Felismina Mendes. *Handbook of Research on Health Systems and Organizations for an Aging Society*. January 2020. Accepted: 2020-02-26T14:42:32Z.

[45] Sukumar Ganapati. Florida International University. page 44.

[46] Ricardo Gomes, Ana Ferreira, Luís Azevedo, Rui Costa Neto, Laura Aelenei, and Carlos Silva. Retrofit measures evaluation considering thermal comfort using building energy simulation: two Lisbon households. *Advances in Building Energy Research*, 0(0):1–24, September 2018. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/17512549.2018.1520646.

[47] Lorenz Hurni. Multimedia Atlas Information Systems. In Shashi Shekhar and Hui Xiong, editors, *Encyclopedia of GIS*, pages 759–763. Springer US, Boston, MA, 2008.

[48] J. Johansson, T. Opach, E. Glaas, T. Neset, C. Navarra, B. Linnér, and J. K. Rød. VisAdapt: A Visualization Tool to Support Climate Change Adaptation. *IEEE Computer Graphics and Applications*, 37(2):54–65, March 2017. Conference Name: IEEE Computer Graphics and Applications.

[49] Robert Johnston and Panupak Pongatichat. Managing the tension between performance measurement and strategy: coping strategies. *International Journal of Operations & Production Management*, 28(10):941–967, January 2008. Publisher: Emerald Group Publishing Limited.

[50] JS Foundation js.foundation. jQuery.

[51] Robert S. Kaplan and David P. Norton. Transforming the Balanced Scorecard from Performance Measurement to Strategic Management: Part I. *Accounting Horizons*, 15(1):87–104, March 2001. Publisher: Allen Press.

[52] W. R. Keatinge, S. R. K. Coleshaw, and J. Holmes. Changes in seasonal mortalities with improvement in home heating in England and Wales from 1964 to 1984. *International Journal of Biometeorology*, 33(2):71–76, June 1989.

[53] Mary Pat MacKinnon and Judy Watling. Executive Summary. page 3.

[54] Miguel López Mamani. What is Vue.js and How do we Use It?

[55] Ricardo Matheus, Marijn Janssen, and Devender Maheshwari. Data science empowering the public: Data-driven dashboards for transparent and accountable decision-making in smart cities. *Government Information Quarterly*, 37(3):101284, July 2020.

[56] Andrew McAfee and Erik Brynjolfsson. Big Data: The Management Revolution. page 9.

[57] Ana Monteiro, Vânia Carvalho, Teresa Oliveira, and Carlos Sousa. Excess mortality and morbidity during the July 2006 heat wave in Porto, Portugal. *International Journal of Biometeorology*, 57(1):155–167, January 2013.

[58] Claudia Sousa Monteiro, Carlos Costa, André Pina, Maribel Y. Santos, and Paulo Ferrão. An urban building database (UBD) supporting a smart city information system. *Energy and Buildings*, 158:244–260, January 2018.

[59] Zan Azma Nasruddin, Azleen Markom, Maslina Abdul Aziz, and Pa'ezah Hamzah. Evaluating construction defect mobile application using think aloud / Zan Azma Nasruddin … [et al.]. *Malaysian Journal of Computing (MJoC)*, 3(2):162–171, 2018. Number: 2 Publisher: Universiti Teknologi MARA Press (Penerbit UiTM).

[60] Pape Abdou Ndour. Environmental impact of natural resources exploitation in Nigeria and the way forward – University of Maidugury.

[61] Tina-Simone Neset, Tomasz Opach, Peter Lion, Anna Lilja, and Jimmy Johansson. Map-Based Web Tools Supporting Climate Change Adaptation. *The Professional Geographer*, 68(1):103–114, January 2016. Publisher: Routledge _eprint: https://doi.org/10.1080/00330124.2015.1033670.

[62] Jakob Nielsen. Estimating the number of subjects needed for a thinking aloud test. *International Journal of Human-Computer Studies*, 41(3):385–397, September 1994.

[63] P. J. Nogueira, A. Machado, E. Rodrigues, B. Nunes, L. Sousa, M. Jacinto, A. Ferreira, J. M. Falcão, and P. Ferrinho. The new automated daily mortality surveillance system in Portugal. *Eurosurveillance*, 15(13):19529, April 2010. Publisher: European Centre for Disease Prevention and Control.

[64] Ana Raquel Nunes. The contribution of assets to adaptation to extreme temperatures among older adults. *PLOS ONE*, 13(11):e0208121, November 2018. Publisher: Public Library of Science.

[65] Rosa Oliveira, C. António, Hugo Santos, and E. Fernandes. The portuguese housing stock: an initial analysis of the national energy certification database, 2017.

[66] Tomasz Opach and Jan Ketil Rød. Cartographic Visualization of Vulnerability to Natural Hazards. *Cartographica The International Journal for Geographic Information and Geovisualization*, 48:113–125, June 2013.

[67] C. Rooney, A. J. McMichael, R. S. Kovats, and M. P. Coleman. Excess mortality in England and Wales, and in Greater London, during the 1995 heatwave. *Journal of Epidemiology & Community Health*, 52(8):482–486, August 1998. Publisher: BMJ Publishing Group Ltd.

[68] Susana Pereira Silva, Inês Batista, Liliana Antunes, and Baltazar Nunes. Estimativas do excesso de mortalidade associado a períodos de calor extremo ocorridos em Portugal em 2014. report, Instituto Nacional de Saúde Doutor Ricardo Jorge, IP, November 2014. Accepted: 2014-11-26T15:50:29Z.

[69] Susana Pereira Silva, Ana Rita Torres, Ana Paula Rodrigues, Mariana Neto, Carlos Matias Dias, Silvia Antunes, Jorge Marques, and Baltazar Nunes. FRIESA (FRIo Extremo na SAúde): relatório da época de inverno 2018/19. report, Instituto Nacional de Saúde Doutor Ricardo Jorge, IP, June 2019. Accepted: 2020-06-03T18:12:48Z.

[70] Liliana Sousa, Helena Galante, and Daniela Figueiredo. Qualidade de vida e bem-estar dos idosos: um estudo exploratório na população portuguesa. *Revista de Saúde Pública*, 37:364–371, June 2003. Publisher: Faculdade de Saúde Pública da Universidade de São Paulo.

[71] Ogan M. Yigitbasioglu and Oana Velcu-Laitinen. The Use of Dashboards in Performance Management: Evidence from Sales Managers. *The International Journal of Digital Accounting Research*, 12:36–58, 2012.