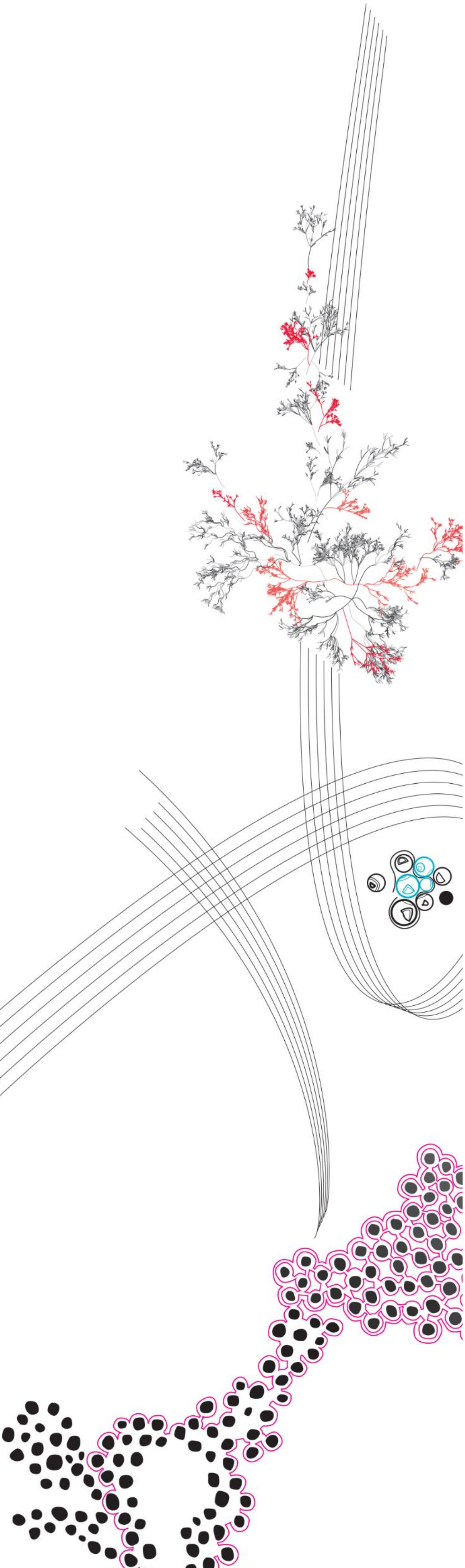


Air Temperature Visualization of Public Spaces in Enschede

Yoann M.E. Latzer

B.Sc. Thesis (March 2018)

Supervisor: Ir.Ing. R.G.A. Bults

Critical Observer: Ir. J. Scholten (Hans)

Creative Technology

Faculty of Electrical Engineering,

Mathematics and Computer Science

University of Twente, P.O. Box 217

7500 AE Enschede, Nederlands

UNIVERSITY OF TWENTE.

Abstract

Municipalities are and will be confronted with many ecological challenges. Temperature monitoring through the use of Internet of Things sensor networks can provide precise information about the past, current and eventually future environmental situation. The unequal repartition of heat is particularly significant in cities' center, causing an urban heat island effect which can cause infrastructural damages as well as aggravated risks for human's health. The goal of this project is to develop a monitoring solution for a network of temperature sensors scattered throughout the city of Enschede. A complete solution including a web server and web page visualizations were developed to allow the municipality to monitor the intensity of the urban heat island effect in real time. The concept was developed according to the requirements of the different stakeholders using an explorative approach through several idea generation methods. The municipality was enthusiastic about the final prototype and believed that the visualizations could help analyzing and sharing the information gathered.

Acknowledgment

I would like to thank my supervisor Mr. Bults for his crucial help and guidance and my critical observer Mr. Scholten for his helpful feedback during the entire project. I would also like to thank Mr. Teekens and Mr. Meijer from the municipality of Enschede for their cooperation and feedback.

Additionally I would like to thank my partner, Tom Onderwater, for his help and overall professionalism. The final prototype could not be operational without our close collaboration.

During these three years I encountered some of the most motivating teachers and staff members, and I would also like to thank all of them for building the Creative Technology Bachelor every day. Finally I would like to thank my parents for believing in me and for their support during my entire Bachelor.

Table of content

List of figures:

Appendices

1. Introduction

The smart city vision defined by Zanella et al. aims at using the latest information and communication technology to support added-value services for the city administration and its citizens[1]. In accordance with this vision, the municipality of Enschede is looking for innovative solutions to monitor and visualize the air temperature levels in the city. This project is conducted by Tom Onderwater and Yoann M.E. Latzer, respectively responsible for the hardware conception along with its deployment and the end-user user software, including data collection and distribution through an online server.

1.1 Client and context

Enschede is a city and a municipality located in the eastern point of the Netherlands in the province of Overijssel and in the Twente region. It features an oceanic climate like most of the country, and is composed of various landscapes ranging from residential urban areas to rural farmlands. According to the KMNI (Koninklijk Nederlands Meteorologisch Instituut) the city recorded temperatures as high as 36.1 °C during the summer between 1981 and 2010[2]. In this context the goal of this graduation project is to build an internet of things network consisting of air temperature monitoring systems which will communicate their measurements to an online interface.

1.2 Challenges

The municipality of Enschede, like most cities, faces a heat buildup in its most urban areas (e.g.: commercial zones in the city center, urban residential areas) when the temperatures are particularly high. This is caused by several parameters, mostly architectural ones, and known as the urban heat island effect. Therefore certain parts of the city might be exposed to higher health risks which already exists when experiencing high temperatures.

This effect is very hard to avoid entirely, mainly because it would imply long term architectural changes, but it can be measured in order to understand its intensity and eventually take preventive actions (e.g.: dehydration warning for citizens living in zones that are greatly affected). Such measurements can be performed using air temperature sensors placed strategically in different areas of the city.

But gathering such data to monitor the intensity of the urban heat island effect is not enough. A graphical interface must be provided to assure that the municipality can easily monitor the entire system and analyze its data. Because such a system can be useful for the municipality to make data-driven decision, it must provide an overview of the current situation but also allow for in depth analysis. A simple representation of this system is introduced on figure 1.

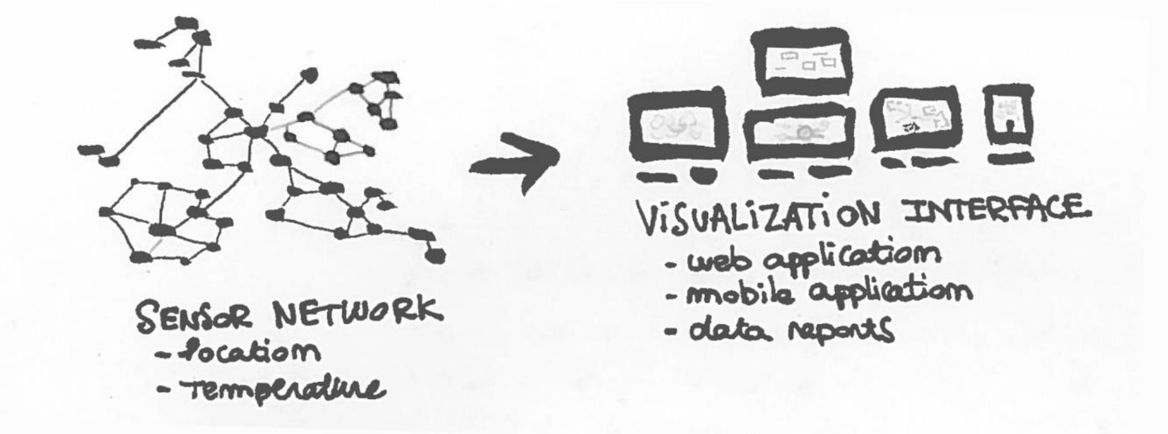


Figure 1: Sketch of a simplified system overview

1.3 Research questions

In order to solve the air temperature visualization challenges and produce an adequate solution to the municipality the following research question and sub-questions were proposed:

How to format and visualize air temperature measurements of public space in the city of Enschede for administrative and public use?

> *What are the possible use cases of air temperature measurements for the identified stakeholders?*
>
> *How to provide an overview and useful insights of the measured air temperatures to the different stakeholders?*

These questions are focusing on the efficiency and relevance of the final interface. By emphasizing on this aspect the current project aims at determining which specific issues data visualization could help fixing, who could benefit from using an interface providing such data and which technical and graphical rules must be used to do so.

2. State of the Art on Air Temperature Visualization

There are very few academic literature focusing specifically on visualizing air temperature, therefore in order to gain insights of the state of the art this chapter is divided in four parts. Firstly the effects of high temperatures will be briefly introduced in order to understand what our final system may have to help preventing or detecting. Then the general advantages of representing data through a visual medium are discussed, along with preferred type of environmental data visualization. Thirdly the advantages of gathering data through an internet of things network will be presented to emphasize on the potential administrative decisions our interface may help to influence. Finally this chapter will introduce several existing solutions, which includes air temperature measurements, in order to understand how and why such systems are implemented.

2.1 The effects of high temperature

This section will briefly describe the urban island effect before listing common effects of high temperature on health.

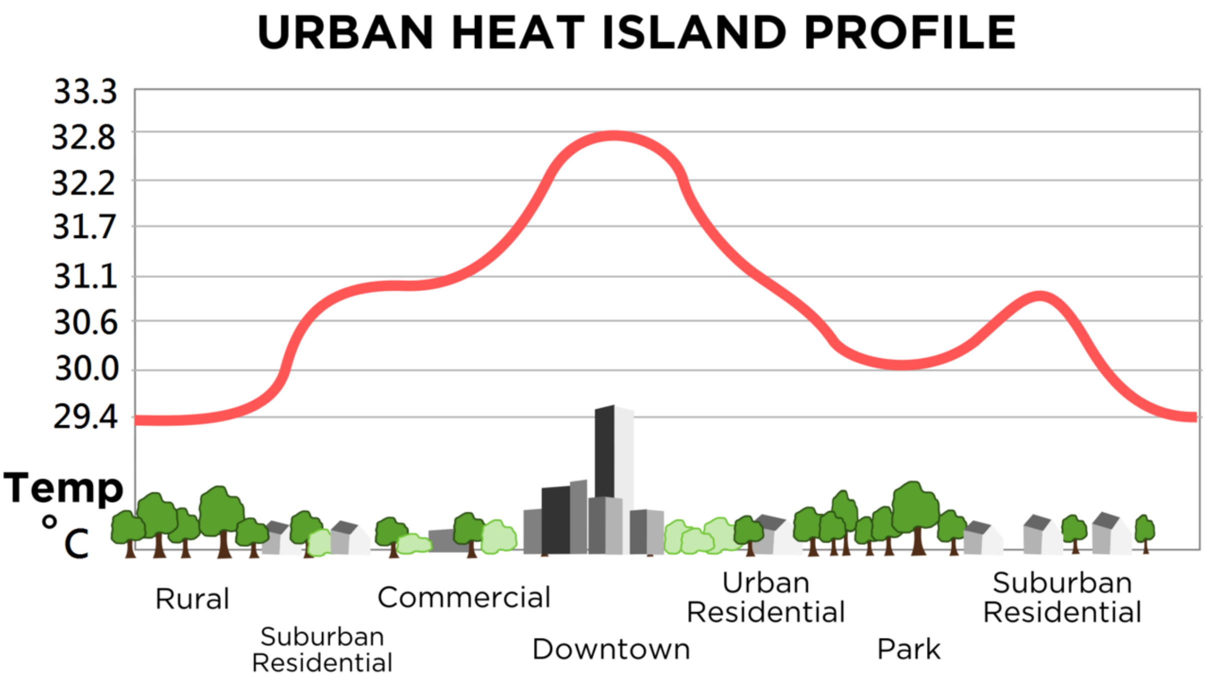


Figure 2: Graphical representation of the urban heat island effect

The distribution of heat in urban areas differ from rural areas with the most apparent effect in temperature distribution being called an urban heat island [3], as represented above on figure 2. The main cause for this effect are architectural parameters (typically the use of bricks, concrete, asphalt and similar material which absorb more short-wave solar radiation) as well as pollution level parameters caused by the city’s activity. The conditions for such an effect are defined by William D. Solecki et al. [4] as “heightened air and surface temperatures in urban areas relative to surrounding suburban and exurban areas” and it is noted that while the effect can occurs throughout the year it becomes a public policy concern in the summer because of its co-incidence with heat waves.

As reported by H.L. Macintyre et al. [3] commons effects of high temperatures on health includes heat exhaustion, heatstroke and emergency hospitalizations, which can all lead to death in the most extreme cases. These effects are amplified by different parameters such as the types of housings or the age of individual citizens. While deeply understanding how to prevent such negatives effects is out of the scope of this literature review, it is important to note that our solution may provide the municipality crucial insights than can be communicated to health services.

When temperature levels are raising the heat is distributed differently in urban areas. Such amplified heat levels can become a serious threat for health in the most affected areas.

2.2 The advantages of visualizing data

The following section will explore how humans tend to process, understand and remember visual information better than other forms of data representation. Additionally, we will see that in the context of analyzing environmental data, maps are powerful tools to gain insights about the evolution over time and space of such parameters.

The idea that “a picture is worth a thousand words” is not new and relies on the fact that humans are visual-oriented species. It seems that humans values images because they carry a story which allows for faster information processing. We do not only understand information faster but it also stays longer in our memory [5]. This is the reason for Siricharoen et al. [5] to suggest that visual communication is an efficient tool to ease the distribution of information through social medias. Wei Lu et al. [6] goes further and indicates that the visual data representation also allows for deeper insights when analyzing vast amounts of measurements. This is partly due to the fact that it can help noticing spatio-temporal patterns when made dynamic. While agreeing on the processing efficiency and impact related advantages, Waralak V. Siricharoen et al. [7] suggest that there are some limitations to infographic representations. First of all data can be skewed or have a large margin of error making it irrelevant when it is visually presented. Secondly displaying too much information might also cancel the benefits of visualization, even though this problem can be solved with interactive infographics which allow to focus on a smaller portions of large datasets [7].

When looking at temperature visualization related literature, spatial-based representation of data seems to be a particularly useful tool to analyze such environmental data. Jan Hjort et al. [8] notes that maps can be used for efficient analysis and prediction of air temperature measurements when combined with data processed according to the statistical generalized additive model. Bin Shao et al. [9] indicates that in the context of visualizing heat island effects the three dimension based models are more authentic than the two dimensional representation. Furthermore, Brooke Fisher Liu et al. [10] states that presenting information in a map increases the understanding and potential compliance with recommended actions in warnings. Both acknowledge that representation data using a map is preferred when extracting and presenting information from datasets in their respective field of research.

Giving a visual output has many advantages but also certain limitations. It can be useful for both communicating information to a large audience and helping experts to analyze certain datasets. In the context of environmental measurements, maps are preferred when analyzing the evolution of different parameters (e.g.: air temperature, air quality) over time and space.

2.3 Advantages of internet of things systems for smart cities

In this section the advantages of internet of things systems are introduced in order to anticipate the type of data that will need to be processed for this project. These systems are commonly called “smart cities” when a municipality is using the most advanced information and communication technology in order to make its traditional networks and services more flexible, efficient and sustainable [11].

An important aspect of smart cities is the possibility to make data driven decisions for “higher resource utilization, reduced capital and operational cost structure, risk identification and management, and sustainability” [12]. This is made possible with sensors connected in an internet of things network, making automatic measurements in real time without the need for human operators to intervene. Such networks can provide live measurements (e.g.: of air temperature or humidity) which can be used to build a database of building’s structural integrity over time, allowing targeted and proactive maintenance at a lower cost [1].

These solutions can also help monitoring other environmental parameters such as the air quality but also the overall energy needs of different services (e.g.: public lightning or transportation). It provides the municipality a clear and detailed overview of these parameters [1]. These technological and organizational improvements are recognized in research as key factors to improve sustainability in general terms [13], with empirical results proving that the reduction of CO_2 emissions are achieved "once a threshold level of ICT development has been achieved" [14].

In order to fully exploit the advantages of an internet of things network it is important to aim for helping the municipality to make data driven decisions. Therefore in the context of this project it means that making all the information gathered by the network of temperature monitoring systems easily available for decision making is crucial. However the range and nature of such decision is dependent on the specific demands from the client, which will be determined in chapter 3.

2.4 Case study: Detecting Forest Fires using Wireless Sensor Networks

The following section will present a complete system which was implemented in the past. While it does not focus on the repercussions of air temperature level, it does use such measurements to help preventing forest fires.

DIMAP-FactorLink is a Spanish company specialized in various technical solutions but more specifically involved in internet of things systems implementation. In 2010 they released a report on a fire detection system built for SISVIA Vigilancia y Seguimiento Ambiental, covering 210 hectares of forest using wireless sensors [15].

The network of sensors was built using 90 Waspmotes microcontrollers [15], measuring four parameters (air temperature, relative humidity, carbon monoxide and carbon dioxide) and sharing data every five minutes. The choice for this specific hardware was motivated by its energy consumption efficiency as the report notes that each sensor power level is adjustable with most of them set in a sleeping mode to accentuate that efficiency. The overall system can be seen on figure 3.

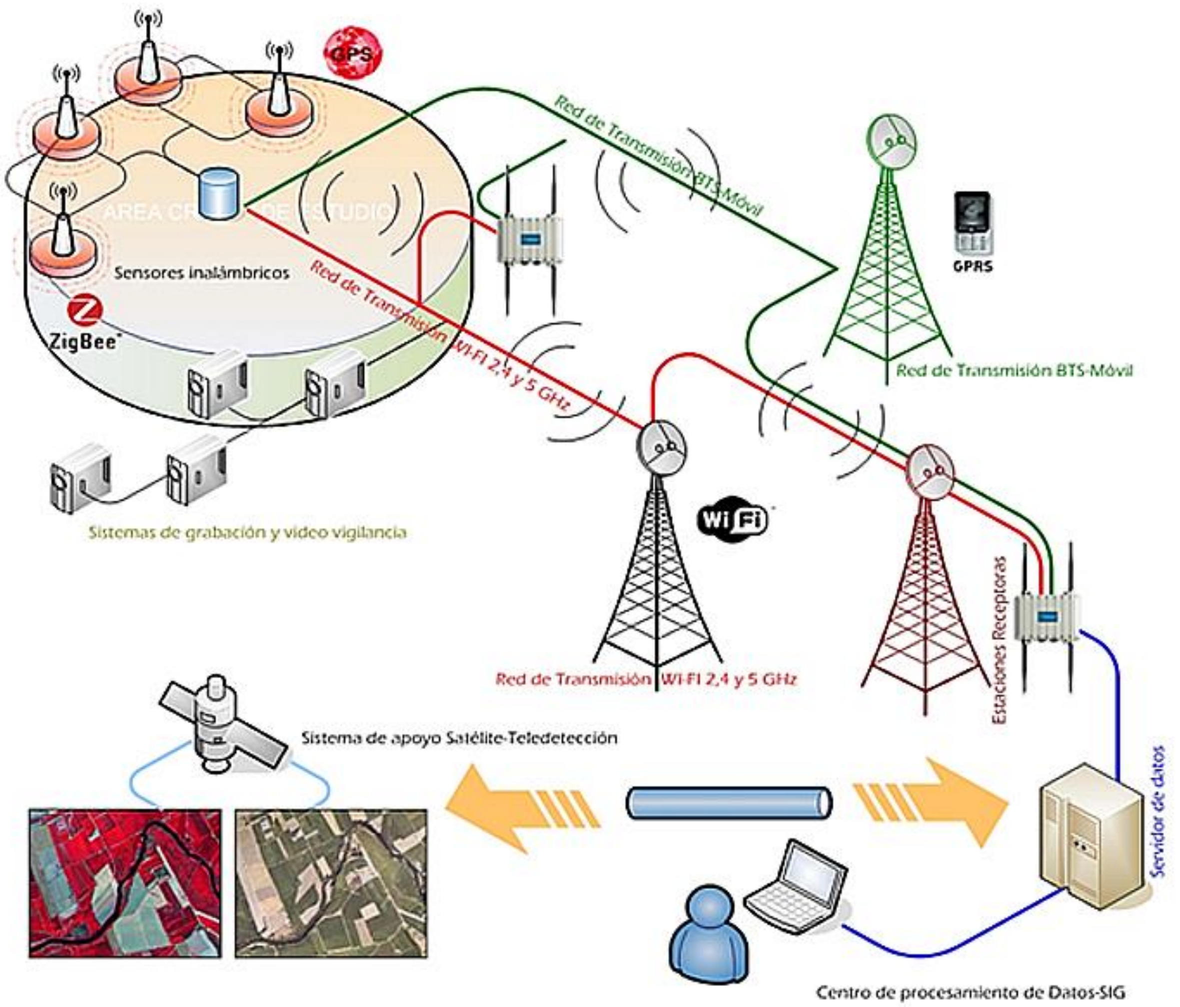


Figure 3: Overview of DIMAP-FactorLink's solution network

The end product includes a live monitoring interface which represents the location and measurements of the sensors on a map (figure 4). This interface also includes other visualization tools such as a panel which represent the location of the sensors along with a table that provides raw data (e.g.: status or last measurement value) for each of them. The system can also include alert managements which can deliver early warning alarms when the measured parameters go above a configured threshold.

Overall this system provides a complete monitoring tool which allows for an overview of the area's environmental parameters as well as detailed information which can be used for further investigation (e.g.: comparison over large period of time). The alarm system also shows the automation possibilities as it can detect abnormal activities by itself.

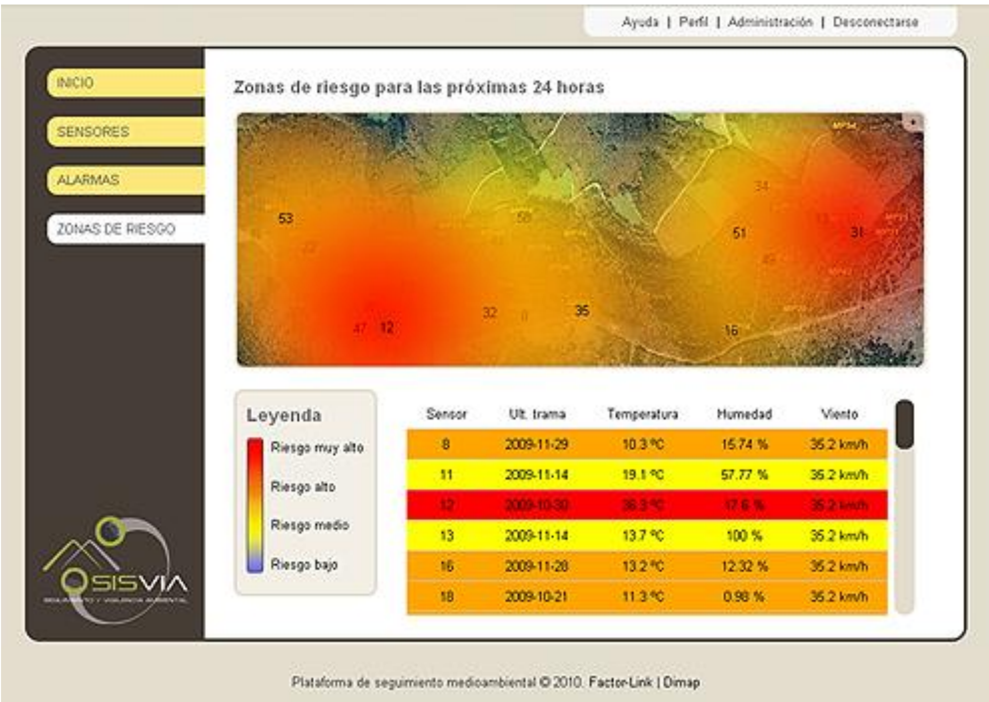


Figure 4: DIMAP-FactorLink's solution final user interface

This system is a one of the few complete solutions which uses air temperature measurements as one of its most important parameters. It is a good example of a solution that provides an overview of a large dataset with constant live updates. It is important to note the use of a map which allows to quickly visualize the current overall situation. A map is also used to easily access a specific sensor and get more detailed information. This is a concrete example of the accessibility of such data representation as we can notice that there is something urgent on figure 4, without the need to understand the entire context.

2.5 Additional examples

Because there are not many examples of fully documented solutions for air temperature level monitoring, partially related projects are briefly discussed in the following section.

Ekobus Project by Belgrade and Pancevo (Serbia) with Ericsson

In 2012 the Serbian cities of Belgrade and Pancevo deployed a system in collaboration with Ericson to monitor several environmental parameters across their municipalities [16]. Waspmotes microcontrollers were attached to the roof of the municipalities' busses to provide live dynamic measurements of 6 parameters: air temperature, relative humidity, carbon monoxide (CO), carbon Dioxide (CO2), nitrogen Dioxide (NO2) and GPS location.

Figure 5: Screenshots of Ekobus' monitoring interface

As seen on figure 5 the monitoring interface displays the collected data and allows for a complete overview of the system as well as detailed insights about each sensor node. Furthermore the data produced by the two municipalities can be stored and analyzed in order to evaluate the impact of certain policies on air pollution for example. This project shows the advantages of an interface powered by an internet of things network: it makes it possible to quickly analyze the overall current situation and to follow the developments of certain parameters over time.

As seen in this project, the use of maps is preferred to give an instant feedback, notably about the current location of the sensor nodes since they are moving in this case. The left screenshot featured on figure 5 shows some interactivity with a tooltip being displayed after clicking one of the nodes. This allows the user to intuitively lookup the current situation of a certain location without any extensive search through a database to locate the right sensor node.

Chicago Array of Things

In 2016 the city of Chicago, in partnership with University of Chicago and Argonne National Laboratory, deployed a city-wide network (currently 50 devices across the city with an aim for 500 units by the end of 2018) in order to measure several parameters [17]. It is meant as a "fitness tracker" for the city and gives live information about the air quality, climate and the traffic among other parameters.

The particularity of this project is the initiative of publically sharing the data produced. The municipality is planning to offer a complete online monitoring tool along with free access to the database for the public in order to stimulate data analysis initiatives through the city of Chicago data portal.

While the project does not necessarily emphasize on data visualization, it is meant to provide tools to make it easy for different members of the community to produce an overview of certain issues in the entire city as well as insights about them. The Chicago data portal already publicly shares several datasets about the municipality (see figure 6) and the Array of Things project will add even more detailed datasets to it.

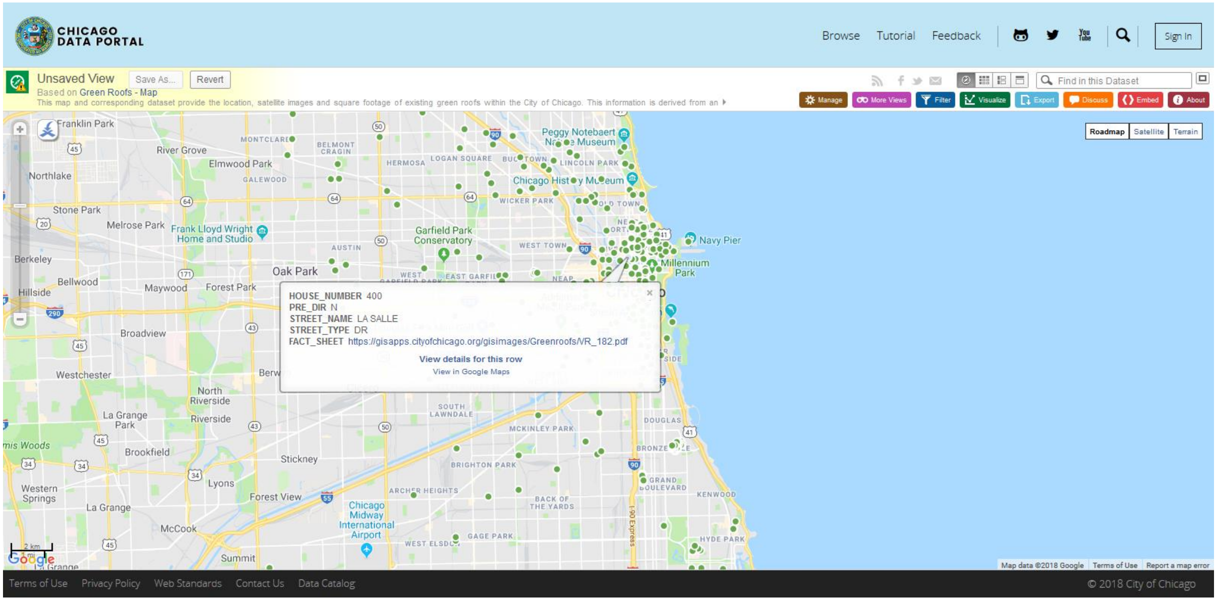


Figure 6: Screenshot of a map representing "green roofs" on the Chicago data portal website

This project shows that a city of the size of Chicago, which is assumed to have quite large administrative and academic staff dedicated to exploiting such datasets, is still trying to reach for its citizens in order to get a different perspective on issues affecting the city. In this context the use of online tools to display large datasets seems to be a way to make complex data analysis available to everyday citizens, where visualization can have a crucial role to raise awareness about issues the administration or academics may have overlooked.

2.6 Conclusion

This chapter reports on the state of the art research performed in an attempt to cover the entire scope of the research questions established for this air temperature visualization project. The effects of high temperature on urban environments as well as on human health were briefly described in order to understand the context from which the data will need to be processed and visualized. The main challenges are detecting the temperature propagation patterns due to the heat island effect as well as determining to which extend the end-user interface could provide interesting information for the municipality to have an impact on its citizen's health.

After noting that visually representing data is an efficient communication tool, we underlined the potential advantages of maps for environmental measurements. While there are many advantages when providing visual information, it can also be a source of error or additional confusion: these elements must be taken into account when selecting a certain scale of type of representation.

Finally the advantages of smart cities powered by internet of things networks were discussed. While visualization can be a tool for fast comprehension, it can also help making data driven decisions when combined with such a network. This implies that the final interface will need to take advantage of both the immediacy of data (e.g.: to provide emergency alerts) as well as its large measurements volume (e.g.: to provide an overview to discover patterns). As seen in the three existing solutions described in the last part, such goals are already partially achieved and proved their relevance for different municipalities.

This state of art research has shown that there are no studies specifically focusing or exploring the different visualization possibilities of an air temperature sensor network. It was found that the established research questions could not be entirely answered by existing literature even though the papers discussed above proves the relevance of such questions. This is partly due to the fact that the focus this project vastly depends on Enschede municipality's specific goals, which are still to be determined at this stage. Hence the need to investigate further the different requirements and stakeholders in order to determine which data and which information we aim to retrieve from the dataset before determining optimal representation methods. This will be addressed the following chapter.

3. Ideation Phase

In order to set solid foundations up to achieve a functional solution that can be later evaluated it is important to summarize the requirements of the project before entering the idea generation phase which will lead to a final idea. This way we can assure that the selected idea will go in the desired direction by settling down on an idea which should be as concise as possible while still describing important details.

3.1 Stakeholders Identification

To provide the municipality with a solution that fits their requirements, the content of this final interface as well as its usability must be regularly discussed and tested with the municipality. Additionally the different stakeholders must be identified in order to produce useful and relevant data analysis and visualization. This is to assure that the final solution will indeed help to gain interesting information for the administration as well as the citizens.

The final interface must include a graphical representation of the measured data and take into consideration two categories of stakeholders: the municipality administration (internal use) and Enschede inhabitants (external use).

3.1.1 Internal use

In this first use case our stakeholders are members of the municipality of Enschede administration. Therefore if they encounter the final interface it is expected that they already have a certain understanding of the context of the project. Furthermore these employees have experience with graphical user interface in order to perform daily office tasks. These two elements indicates that regardless of the personal background of the employees, they will have a certain comfort using our final interface, at least to simply consult it without too much interaction.

But out final interface might involve performing advanced tasks such as exporting a dataset in a certain format or select a certain portion of the dataset. As far as we can tell at this point of the project there is no employee dedicated to exploring the data generated by this project therefore it is important to keep in mind that our interface cannot require hours of learning to be operated. Hence the need for a simple, straight forward interface that can be used quickly and that can also be easily shared among colleagues. The ‘sharing aspect’ is essential to benefit from the data generated by our internet of things prototype, as described in the previous chapter.

3.1.2 External use

Through the very first meetings it was clear that the municipality of Enschede has no intention, at least on the short term, to make the interface available to the public. However it is important to keep in mind future developments and to anticipate a potential public release.

In this case virtually every citizen of Enschede becomes a stakeholder, from the youngest to the oldest. This implies people with both none and advanced experience with computer interface. While this category is too broad to make specific stakeholder requirements, there are still two points that must be taken into account. On one hand the interface must be self-explanatory which means that it must contain enough information for anyone to understand (e.g.: short text to introduce the project, easily readable labels). On the other hand the final prototype should not be able to alter the data set or the interface, or in other words it should not be possible to customize the interface. This is due to public aspect which must imply a certain protection from malicious usages that could compromise the project (e.g.: editing the database with random data), intentionally or not.

3.2 User Requirements

The user requirements can be divided in two aspects: the functional and non-functional requirements. On one hand it is important to produce a technically feasible idea but it is also crucial to keep in mind the final usage scenarios.

3.2.2 Functional requirements

Considering the scope of the project and the usability requirements defined in the previous section, there are several functional requirements involved. Because of the digital nature of the temperature measurements, it is assumed that the final prototype must be a software solution.

First of all, the software prototype must be accessible from a distance since there might be several final users accessing the visualization interface from different machines. Furthermore since we want to visually represent the data, the prototype must be able to both write and read from a database that keeps track of all the measurements made. Finally making requests from this database should be automatic and/or fairly simple on the final interface.

To summarize, our final software prototype must make it possible to easily visualize and manipulate data from a database on a remote computer.

3.2.1 Non-functional Requirements

In order to approach this usability aspect of the final prototype, it is important to elaborate a simple persona of the typical final user and a list of the tasks that he or she will need to perform.

As stated in section 3.1, the final user can be an employee of the municipality or just any citizen. However the final prototype might only be available after a certain amount of experimentation time by the municipality. Therefore our end user will most likely be an employee of Enschede's municipality. We assumed that the end user is fairly used to perform office work related tasks on a computer. This implies that, for example, it could be problematic to assume that the end user is comfortable with console input in order to operate the final prototype. However, interacting with a graphical user interface should be fairly affordable.

The meetings held with the municipality provided a relatively precise list of the tasks the end user will perform:

- Select a node and read its temperature measurements on a visualization
- Adjust the time period that can be seen on the visualization
- Export/download an image of the visualization
- Inspect the measurements present in the database on a table
- Export/download a complete table of the measurements database (e.g.: to perform analysis using an external software)

This list implies that the final prototype provides clear visualizations which can be clicked to display additional information. It should also be easy to change the time period and save the generated visualization in an image. This is crucial to facilitate data sharing as a specific day of measurements could be saved into one simple image.

Furthermore the end user must be able to perform advanced analysis. This means that it should be possible to explore the raw measurements in a table in order to inspect the behavior of a certain node at a certain time. Finally the database should also be available for download in a format that could be loaded on another software for analysis (e.g.: comma separated values, .csv, format).

3.3 Idea Generation

In order to explore and evaluate as much ideas as possible, two methods of idea generation were performed. In each of these the challenges and problems from the first chapter as well as the state of the art and existing systems were taken into account to form a base to generate ideas from.

3.3.1 Visual Idea Generation

The first type of idea generation is purely visual and tries to directly represent, with the use of fast doodles, the final interface. Based on the idea of temperature measurements across the city of Enschede, several types of visualization are tested. This allows for direct, fast feedback which will later serve in the realization while also providing precise ideas for the specification phase. The main results of this phase were drawn on a single A3 sheet which can be seen in the appendix 2.

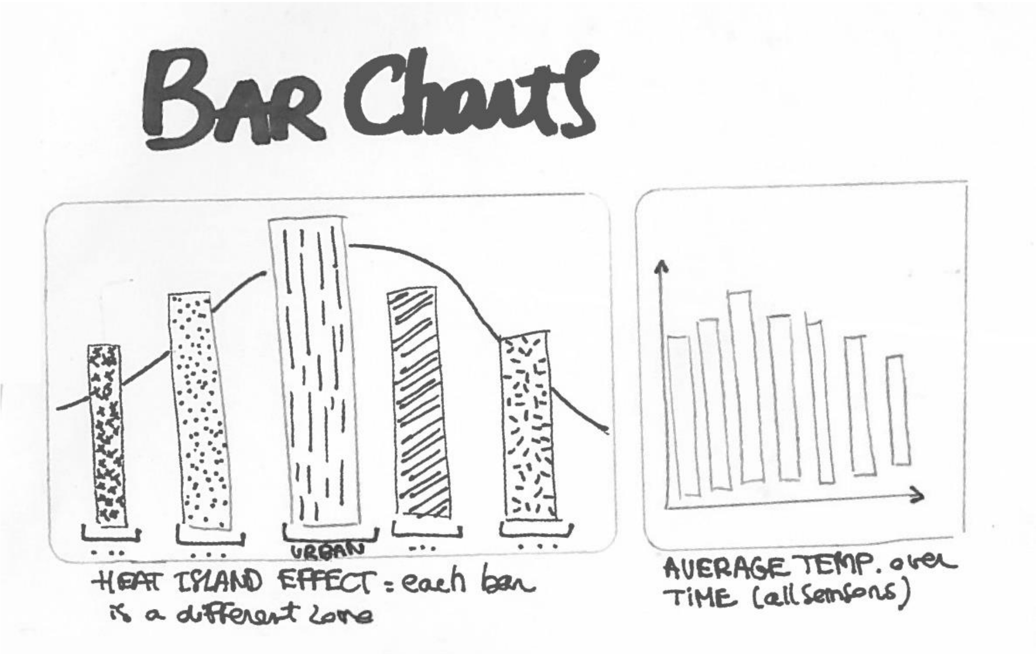


Figure 7: Two different proposal for a bar chart visualization

In the end several representation of the geo-localized measurements were proposed, as well as multiple suggestions for graphical representation (bar charts, bubble charts, etc.). From this first idea generation session it clearly appears that representing the temperature measurements on a map is a must-have since it is both attractive (next to other charts) and very clearly shows the nature of the project. Different ideas were formulated for the temperature representation on the map, as seen on figure 8.

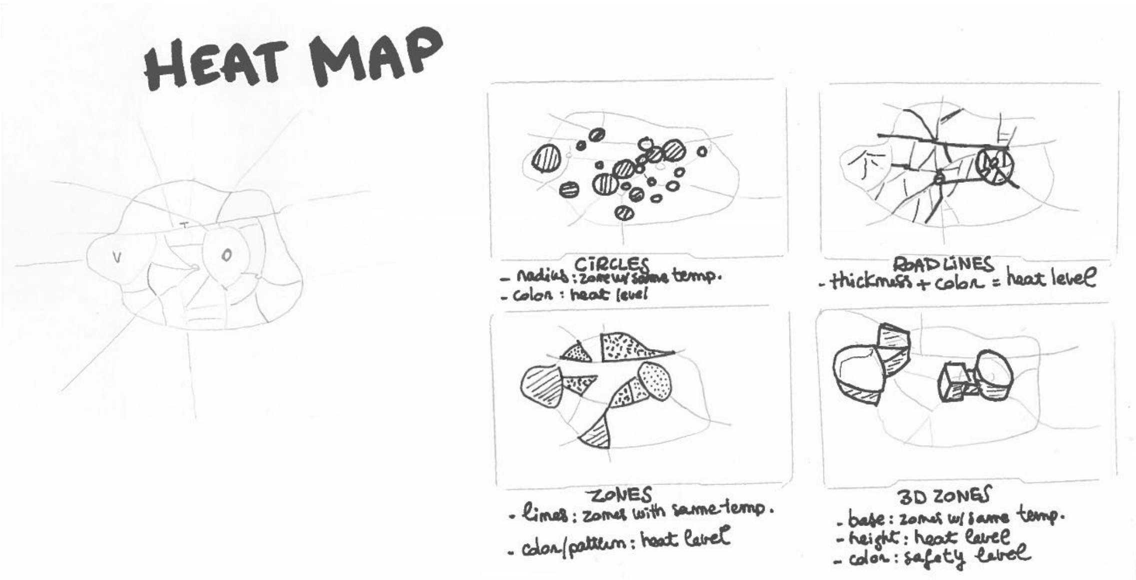


Figure 8: Circles, road lines, zones and 3D zones proposal for representing heat on a map

Some of the ideas generated during this phase were used to build a simple mockup which was presented to the municipality of Enschede. While it is not functional (since we cannot get much information from it), it was an occasion to experience with colors and to contextualize our work for the client. This mockup can be seen on figure 9 and received a positive feedback as its simplicity was appreciated.

Figure 9: First mockup introduced to the municipality

3.3.2 Conceptual Idea Generation

In this second type of idea generation the goal is to detach from the visualization aspect to focus on the different types of application our interface could possibly have. The result is a list of potential requirement the final prototype building could take into account. Afterwards, a first attempt to classify them using the MoSCoW (**M**ust have, **S**hould have, **C**ould have, and **W**on't have) method was performed.

- **M**: Provide a full overview and give insights via "time traveling": the idea is to be able to select a starting and ending time to visualize the portion of the database.

- **M**: Built-in sharing capabilities: the final user can share a part of the dataset or a visualization of it directly on the interface (e.g.: via email or social network account for example).

- **S**: Database export functionalities: either visualized or raw data is available to download in order to exploit the dataset outside of the interface (e.g.: during a meeting for the visualization of with specialists for the raw data)

- **S**: Mobile compatibility: the final interface can also be consulted on a mobile device, possibly with additional functionalities such as consulting the closest node depending on users' location.

- **C**: Display city-wide temperature levels: from a defined amount of sensors, the interface is capable of computing the level of temperature in the entire city.

- **C**: Alert setup system: in order to schedule automatic messages, the final user should be able to define certain rules (e.g.: temperature limit) on the interface.

- **C**: Export complex visualizations such as posters that can be directly used on the city's billboards to inform citizens.

- **C**: Contribute to the database: in this case the, mostly advanced, user is able to record his own measuring device on the interface in order to contribute to the temperature data collection.

- **W**: Style customization: certain elements of the visualization are determined directly by the user, for example the color or size of certain elements can be freely edited.

While certain ideas listed here are clearly part of the defined requirements, other such as the last item would require a complete different scope for the project. However the collaborative aspect is indeed in the scope of the smart city vision, which makes it a possible candidate for future recommendations. These propositions will be reformulated and classified in chapter 4 with the addition of justification for their classification.

3.4 Final Idea

A final idea was derived from the idea generation process described above. The final solution must be a web application which can both record and serve measurements in/from a database, providing an easily accessible interface which includes visualizations. An overview of such a system can be seen on figure 10.

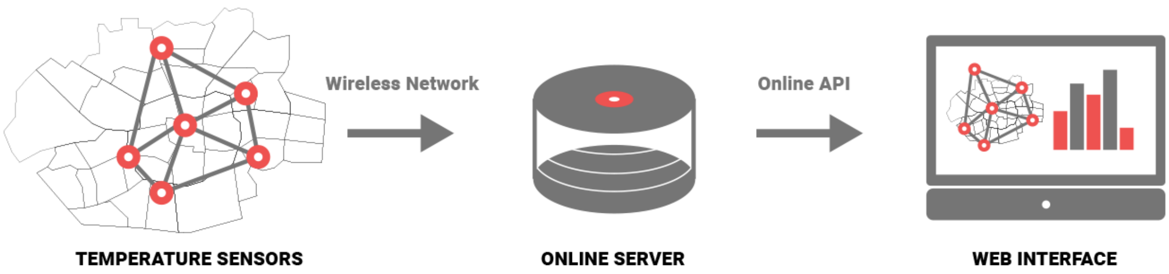


Figure 10: Full system overview

While it can be made responsive to fit different screen sizes, mobile compatibility is not a priority. The web nature of the prototype is chosen in order to provide remote access of the database. The back-end server will gather the measurements and serve them to a front-end which will allow raw or visualized data exports.

The visualization will include a map which will provide an easy overview of the sensor's location along with its latest measurement. The second part of the visualization will include two charts, chosen for their clarity: a bar chart which will represent the heat island effect and a line graph which will represent an overview over time of a selected sensor. Both of these graphs must be easily editable to display the desired part of the database (e.g.: last month's temperature for sensor number 2). Propositions for the arrangement of the selected visualizations were generated during the visual idea generation and can be seen on figure 11.

Finally a part of the interface will make it possible to export a certain part of the database in several formats, such a pdf or csv.

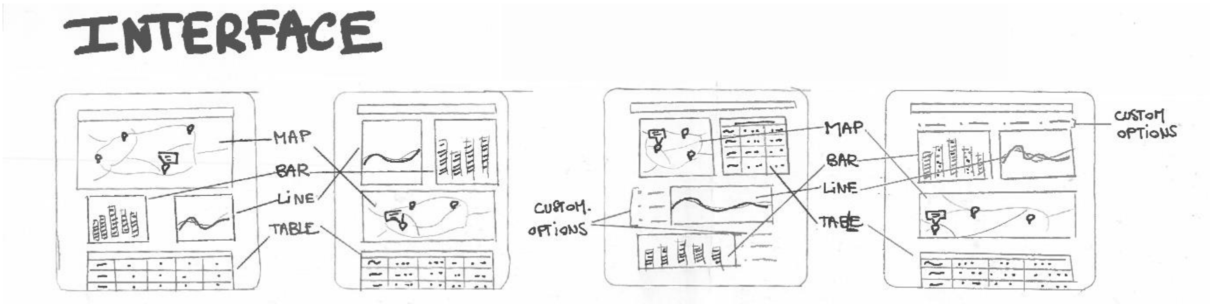


Figure 11: Four different arrangement propositions for the final interface

4. Specification Phase

During the specification phase the idea formulated in the previous chapter and the usage scenario of the final prototype idea are described with more details. The goal of this chapter is to provide precise directions for the realization phase by sorting out the different functionalities depending on their importance, in order to identify crucial characteristics and optional ones.

4.1 User Experience Specification

The goal of this section is to define a precise user experience in order to underline all the functionalities of the final prototype starting from the final user point of view. Providing details on the concrete utilization will allow us to precisely determine the characteristics our prototype should or should not include.

4.1.1 Use Scenario: Consulting Data

Considering the data gathering and delivering aspect of our final idea, the first action that can be taken into account is the consultation of data. The final user will be able to browse through the temperature measurements with two different approaches: consult the latest data available or consult previously recorded data.

In the first case the user interaction can be described with the following ordered list:

1. Connect to/access the interface via a web browser
2. Immediately see on the map the current location and latest measurements of the sensors
3. Consult additional charts for a quick overview over time of the latest measurements
4. Scroll down to seek the raw data tables in order to inspect a specific node and its measurements history

In order to consult previously recorded data, the user scenario slightly differ from the one described above in this section. In can be described with the following ordered list:

1. Connect to/access the interface via a web browser
2. Focus on the desired visualization (or raw data table) and identify a time selection tool
3. Input the desired time period (starting and ending date)
4. Select all nodes or a specific set of nodes (or a single one)
5. Apply the time selection by clicking a button

6. Consult the temperature measurements for a certain period of time

In both cases the user is simply reading information displayed on the screen in order to either make sure that the system is still properly operating (e.g.: verify that all the nodes produced data recently) or to inspect temperature measurements at a given period of time. This can be useful for quick verifications but in order to exploit (share or present) the dataset we need to take into account data export in a second use scenario.

4.1.2 Use Scenario: Exporting Data

As seen in the first chapter, the municipality of Enschede is seeking for an innovative solution which implies that this temperature measurement system can be seen as an experimentation. This implies that the result of such experiment must be made easy to share, in order to provoke constructive feedback for future developments. In our use scenario this is represented by exporting the database (or a part of it).

While exporting data will give a different result with visualization or row tables, the user interaction will stay relatively similar. The data is then downloaded as the appropriate format (e.g.: png for images and csv for data tables).

This scenario is similar to the second one described in section 4.1.1. The only difference is that the final user clicks a button labeled “export” after selecting a certain amount of nodes and a time period. This button can be either next to a visualization to export an image, or next to a table to export raw data.

4.2 Project Requirements

In this section the specification will be ordered and sorted using the MoSCoW method which will determine the must-have, should-have, could-have and won't-have features. These functionalities are presented in the following table before discussing the choice for sure qualification.

Must-have	**S**hould-have	**C**ould-have	**W**on't-have
M1. Gather and store data measured by the sensor	**S1**. Gather data without interruption	**C1**. Display temperature for entire city zones	**W1**. Possibility for users to contribute to the dataset
M2. Display a map with measurement points	**S2**. Select a certain number of sensor(s) to display	**C2**. Responsive design (tablet and mobile compatibility)	**W2**. Ability to edit the style of the visualizations
M3. Represent the heat island effect on a chart	**S3**. Export raw data tables	**C3**. Display a quick overview of the measurement on a chart	
M4. Select a certain time period to display	**S4**. Include a secured login function		
M5. Export images of the visualization			

Figure 12: Project Requirements MoSCoW Table

The very first requirement is M1, which is at the core of my collaboration with Tom Onderwater, represents the storage of measured data which is crucial to make it available for the visualizations. M2 and M3 are the main functionalities of the interface, with two different goals. M2 will provide a quick overview which is very useful for the final user to picture what the full system looks like on the ground, while M3 will take into account the position of the sensors in order to provide information on the heat island effect, which was introduced in section 2.1. Finally, M4 and M5 are crucial in order to make the data exploitable for the municipality of Enschede, as described in the previous section of this chapter.

In the should-have column, elements that would preferably be delivered to the municipality are listed, even though their absence would not compromise the final goal of this project. S1 is classified in this category because if the data gathering system fails to collect information for a short amount of time, the database will still be relevant. However it is preferable to avoid any interruption even though the system cannot be constantly online when being developed. S2 is also not crucial to exploit the data, but it could provide the final user a useful option when searching for insights in the dataset. S3 would be very useful for data analysis but as stated above the municipality does not specifically seek for such an advanced use of the data, which makes it a should-have functionality. The absence S4 would again not compromise the entire project but it would assure the municipality that their project is safe from public attention which can be useful to create an experimentation mindset.

After should-haves come the could-haves which describe elements that would make the overall experience more comfortable or interesting for the final user but are not crucial in any way. C1 is the most ambitious one, as it implies that the final interface is capable of projecting temperatures levels in a certain area around the sensor. This implies complex computation which could possibly fall outside of the scope of this project. While the final interface should be readable on most desktop computers, C2 suggest that it would also possible to consult and use the final prototype on a mobile device, which could be comfortable or useful for certain final users. Finally C3 represents an additional chart idea which would help providing an overview of the system over time without any user input.

To conclude the requirements section, two elements are listed which lies outside of the scope of this project. W1 suggest that the user could contribute to the database, with its own sensor setup for example. This would require a platform which can accept new devices and verify that the data sent is accurate. Additionally W2 suggests that users could change the style of the visualization as they wish, which would require to build an entire selection of color palettes, in order to avoid unreadable charts output. While such functionalities could be implemented in the future, it cannot fit with the time constrains of this project.

4.3 Functional Specifications

In order to conclude the specification phase this section will describe the functional specifications for each of the elements of our final prototype, taking into account the information previously stated in this chapter. The final interface idea will be separated into four sections which represent four main functionalities: gathering data, delivering data, visualizing data and selecting data to consult or export.

4.3.1 Data Gathering

As described in the previous section, collecting temperature measurements is a crucial element to build the final visualization prototype. As specified during the ideation phase, such measurements are digital and must be available at any time from different locations.

These two elements make the choice of a web server obvious, in order to be able to collect the temperature measurements pushed to the internet by the sensors in a database and to make it available at any time.

The web server will ideally be constantly online in order to avoid missing any measurements and to make them available at any time. This last step is described in the following section.

4.3.2 Data Delivering

As the data is gathered in an online database, the next logical step is to deliver such data to the final user. This is the second task of the web server: deliver, on client demand, the database for the final user to consult.

In order to assure a comfortable experience the delivery of the web page must not take too much time, which considering the average modern web browsing experience cannot be more than a minute at the very maximum. Ideally the data delivering takes less than a few seconds in order to make this step almost invisible for the final user to focus on the operations described in the following section.

Finally the data delivering should assure that the integrity of the database is delivered, which means that all the different final users can access the exact same database.

4.3.3 Data Visualization

In this section the specification for the main tasks that the final user will perform are described. The two previous sections are describing crucial elements for the interface to function, however they will be completely invisible for the final user. Once the database is delivered from the web server to a web page, it needs to be represented in a visual form in order to make its consultation easy. However it is also interesting to include raw data (e.g.: in a table) under the visualization to allow detailed inspections. Therefore four different visualization methods will be delivered: a map, a heat-island effect chart, a history of the latest measurement chart and a raw data table.

4.3.3.1 Color palette

Before detailing the visualizations and their functionalities, it is important to note that specific direction was taken when choosing the color palette. In most interfaces (e.g.: the DIMAP-FactorLink's solution final interface shown on figure 4) the color represents the level of temperature. For example red is high and blue is low.

Because the main goal of this project is not to evaluate if the air temperature outside is warm or not, this 'color = heat level' rule was broken. This decision was made to put the emphasis on the comparison of levels of temperature based on the location type of the sensor. Therefore warm color were chosen to represent highly active zones (e.g.: red for the city center, orange for sub urban area) and green colors were chosen for rural areas.

This will allow for a clear distinction of the sensors based on their location rather than their current temperature level. To avoid any confusion this color palette will be used across all the visualizations. The chosen colors are displayed in chapter 5.

4.3.3.2 Map Visualization

The map will simply show the latest position of the sensors along with their respective latest measurements. This map should by default fit all the different sensors in order to provide a quick overview of the system, but the user should also be able to freely move it in order to facilitate orientation (e.g.: if the position is not clear, the user will be able to browse the surroundings and eventually identify a place he/she knows). Finally, the final user must be able to click on different nodes represented on the map to display their latest measurements.

4.3.3.3 Heat-island Effect Chart Visualization

In order to clearly identify a potential heat-island effect in the city, the system will include sensors placed at strategic points in the city. This chart must represent the nature of these placements (e.g.: is the sensor in an urban area or in a park?) and try to visually display the differences in temperatures in these different zones. This will allow the final user to quickly identify the amplitude of the heat-island effect: in other words it will be easy to see if there are large differences between the different zones of the city or not.

4.3.3.4 Measurements History Visualization

This simple line chart will display the evolution of temperature in the arbitrarily chosen time window. With time on the X-axis and temperature level on the Y-axis it will be easy to visualize the evolution of temperature in the past.

4.3.3.5 Raw Data Table Representation

While a simple table cannot be considered as a data "visualization", it is still listed here since it will be visually present on the final interface. This table should simply show the entire database and should not take too much space on the final interface. This is because this element will only be used in very specific, advanced operations (e.g.: find what a sensor measured on a specific date).

4.3.4 Data Selection and Export

The last step required to make sure that the municipality can experiment with the final interface and take actions based on this experiment is the data selection and export functions.

Data selection implies that the final user will be able to select a certain proportion of the final database. There a two aspects that can be selected in the context of our temperature measurements database: the time period and the amount of nodes. As stated in section 4.2 the ability to select a certain time period is more important that the ability to select which nodes to include. However, this last functionality is still classified as a "should-have" since it would make database browsing fully customable.

In order to share the experimentations and discoveries made with the final interface, it is crucial for the final user to be able to share them. This is made possible with a "export" function, which as seen in section 4.2 must include the possibility to download pictures but should also make the raw data available. This functionality must take into account the dataset selection as described above and it must be straight forward (e.g.: a clearly visible button labelled "export" or "download as an image").

4.3.5 Formatting Data for the Visualization

In this section a simple chart is introduced, which represents how data is formatted in order to be used by the different visualization methods. Figure 13 represents how the data is put in different arrays (represented with a red background) which contains several variables for each of the sensor nodes or each of the measurements.

The first one, labelled 'last_measurements', contains six different arrays which contains the information needed to make both the map and the bar chart. Since it always contains the latest measurement available for a specific node, there is no need to include a timestamp.

The second one, ‘past_50_measurements’ was given an arbitrary value of fifty, which means that it contains the last fifty recorded measurements. This value can be adjusted later, especially when including an option to select a certain time period. It does not contain location information as these are not needed to represent the evolution of temperature over time.

Finally ‘all_measurements’ contains the entire database, with all the information available. It is used to fill a table in order to display raw information for the user to explore.

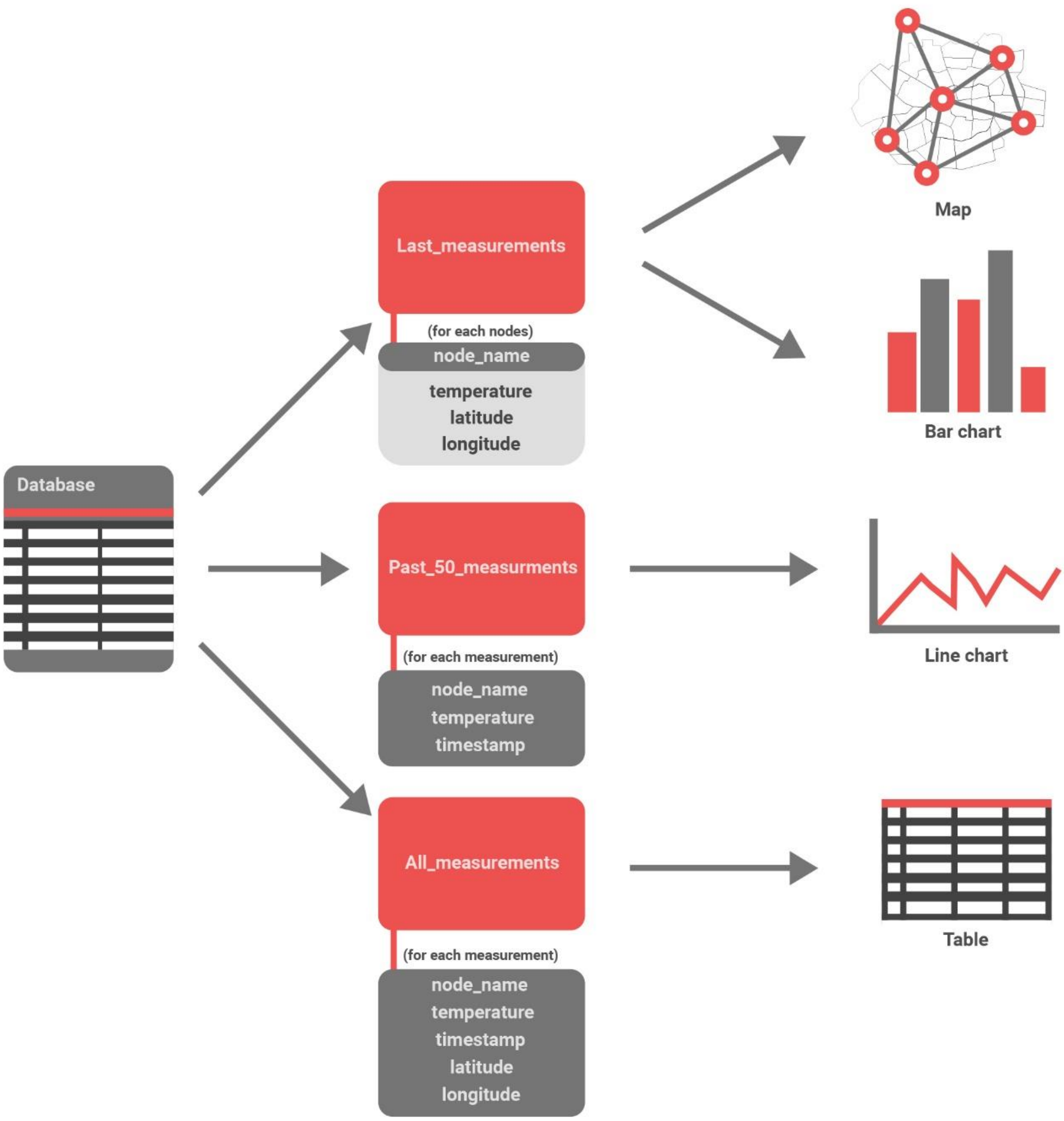


Figure 13: Overview of the data formatting to fit the visualizations

4.3.5 Overall System Architecture

The main functionalities are put together in an overall system architecture which will show the interaction between the main functions of the server. This overview of the full system will be used as a base for the realization phase.

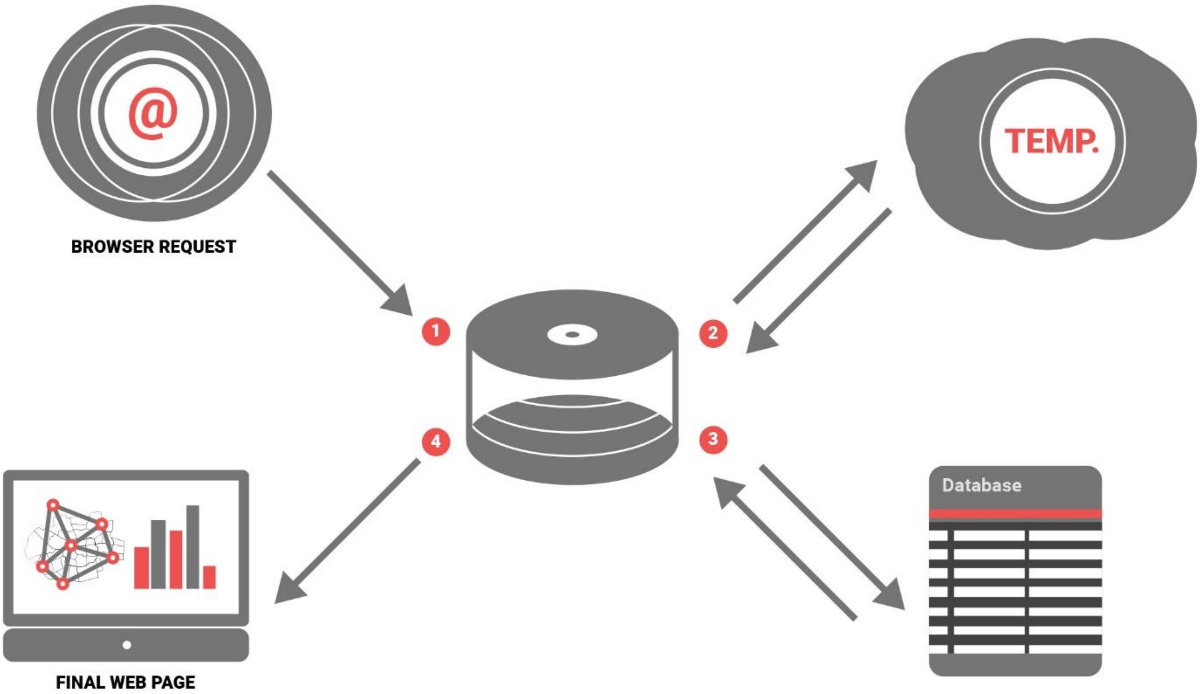


Figure 14: Overall system architecture

The four main functionalities defined in this overview of the server software are labelled on figure 14, which represents the main server in the middle:

1. This part of the system handles the request made from a web browser, at a certain URL. The server must know which information to deliver depending on the request (in our case there is only one main page which contains the visualizations).
2. The server must run a script which connects to the internet, where the sensors will push their measurement. It must run constantly in the background to assure that no measurement is missed.
3. This script should handles two scenarios. First of all it will record every new measurement detected by functionality number 2 in a database. Additionally it must deal with data request from the server.
4. Finally this last part of the system return several elements, according to the request handled by functionality 1. This means that it must send both the web page files and the data from the database to the client.

5. Realization Phase

The goal of the realization phase is to build a prototype according to the requirements specified in the previous chapters. In this section the different tools used to build the final prototype are introduced in three different categories: the hardware, the back-end software and the front-end software. Each category is linked to the other ones and a final section will focus on explaining how they interact in order to deliver a working prototype to the client.

5.1 Server Hardware

As our final prototype is a web application, there are several ways to make it available on the internet: it can be either hosted on a distant server (e.g.: using services such as Amazon Web Services or Heroku) or a local server (e.g.: a computer running from the University of Twente). The second solution was chosen in order to keep a complete overview of the system. While remote services have their advantages, they can also represent a certain long term cost and eventually do not allow the use of certain technology or methods.

5.1.1 Raspberry Pi

The Raspberry Pi platform was chosen since it is relatively affordable, powerful and has a lot of documentation available. The exact model used for this project is a Raspberry Pi 3 Model B, the latest version available which should be powerful enough to host our application but also to run a GNU/Linux distribution which can be used as a comfortable development platform like any other computer (see the full specifications in appendix 3). The Raspberry Pi is running Ubuntu MATE (Ubuntu 16.04.3 LTS with the MATE Desktop Environment 1.16.2) using a 16GB micro-SD card. It is powered with a 5V (2.1A) power supply and is connected to the University of Twente network through the integrated Wi-Fi interface.

Ubuntu was arbitrarily chosen instead of Raspbian (the official distribution available from the Raspberry Pi Foundation) since the first prototype was developed using a Ubuntu live disk on a laptop. While there should not be any compatibility problem between the two GNU/Linux distributions, the comfort of using the same environment was helpful to stay focused on the server software instead of spending time on troubleshooting the operating system.

After installing Ubuntu MATE using “Win32 Disk Imager” (as recommended in the official documentation), a virtual environment was setup on the machine using “virtualenv”. This tool allows to create virtual space in the computer where we can install libraries without affecting the entire operating system: it is useful for development but also to allow several projects to be developed on the same machine in the future.

Once set up, the hardware is ready to run the server software which consist of two main components:

- The back-end which records temperature measurements in a database and handles requests from web browsers.

- The front-end which defines the structure and look of the web pages delivered by the back-end.

5.2 Back-end Technology

There is a growing list of frameworks that can be used to build full stack (back + front-end) web applications, but the choice was restrained since Python was chosen as the programming language due to its compatibility with the Raspberry platform. Python itself had different web framework libraries and Django was chosen to develop our prototype. While different solutions are available (e.g.: FLASK or Pyramid), Django is among the most popular Python web frameworks even though it is relatively old (it was first released in 2005) as we can see on figure 15.

Top in Frameworks · Week beginning Feb 19th 2018

Name	10k	100k	Million	Entire Web
PHP	↓3,240	↓32,907	↓273,719	↓52,080,678
ASP.NET	↑2,226	↓21,188	↑140,295	↓39,753,697
J2EE	↓1,194	↓6,055	↓27,770	↓1,736,157
Ruby on Rails Token	–870	↓4,139	↓14,584	↑591,049
ASP.NET MVC	–719	↑4,072	↑15,594	↓887,710
ASP.NET Ajax	↓660	↓6,978	↓32,417	↓750,422
Ruby on Rails	↓538	↓2,980	↓12,027	↓813,883
Foundation	↓431	↓3,092	↓13,456	↑417,402
Shockwave Flash Embed	–294	↓2,830	↑24,711	↑5,388,229
Classic ASP	↑259	↓3,222	↑21,425	↓2,449,873
Express	–255	↑904	↓3,840	↓205,936
Adobe Dreamweaver	↓250	↑2,822	↓26,667	↓2,088,704
OpenResty	–233	↓1,728	↑16,919	↑1,206,087
Genesis Framework	–219	↓2,716	↓12,417	↑343,481
Heroku Vegur Proxy	–189	↓925	↑3,975	↑180,600
Django CSRF	↓160	–665	–2,115	↑63,557
Adobe ColdFusion	↓155	↓1,400	↑8,568	↓157,072
DAV	↑124	↑1,346	↓8,328	↓641,743
Telerik Controls	↓116	↓1,489	↓7,866	↓167,923
Laravel	↓105	↓747	↓4,954	↑516,983

Figure 15: List of most used web frameworks from builtwith.com

5.2.1 Django Channels

Another reason to choose Django was its official library called Channels, which allows asynchronous tasks and long running connections though WebSockets. The first one is required to be able to run a script which constantly seeks for sensor data updates asynchronously (without blocking the server from delivering web pages) and the second one is necessary in order to update our web page (with new measurements or when the client is requesting older data) in real time without requiring a reload from the client.

The use of 'consumers' is the core concept of Channels: certain part of the code is reacting upon certain events to spread information across the application. For our prototype it will be triggered when we receive new measurements from the sensors: after recording the data in our database a signal is sent to the client connected via the web page to receive the latest measurement.

In other words Channels replaces the original request/response cycle from Django with messages that are sent across channels. In order to use this library two other servers are running on top of our Django server: Redis (to handle the message exchanging across channels) and Daphne (to handle HTTP/WebSockets connections with the outside world). A full overview of the system can be found on figure 16, at the end of this section.

5.2.2 Database structure and functioning

Before presenting the, rather simplistic, database structure we need to mention where the data we want to catch is available. In order to push the data gathered by the temperature sensors on the internet, my partner Tom Onderwater used The Thing Network (TTN), an open source network built for low powered devices with the LoRaWAN protocol. Using this technology several sensors can push online data to a single endpoint: an online TTN application.

However the TTN application does not save the data: therefore we need to talk with the application using the MQTT protocol (which is designed for devices with a limited network bandwidth). In order to do so a Python script was written (which can be consulted in appendix 4) with the use of the paho-mqtt library and the TTN application's access key.

This script will run in the background, detect when there is a new data available and push it into a table called "Measurement". It contains the following fields:

- **measurement_type:** this is manually set to ‘Temperature’ since this is the only data we will record in our project. However there might be other types of measurements in the future.
- **device_id:** the identification number of the device, provided by the metadata from TTN
- **value:** the actual measured temperature value, provided by the metadata from TTN
- **latitude:** geographic information provided by the device, provided by the metadata from TTN
- **longitude:** *same as the previous entry*
- **day:** time information provided by the server since certain sensor do not include this information in the metadata provided
- **month:** *same as above*
- **year:** *same as above*
- **hour:** *same as above*
- **minute:** *same as above*

This is our main table and it contains all the information required to perform data visualization. A second table is also present which simply consist of a list of nodes (six in total in our final prototype). While the first table is used for all the crucial operation, the second one is simply here to easily (and dynamically) return a list of sensors to format the data.

5.2.3 Delivering the final web page

Before describing the technology used to bring the data visualization together on a web page, an overview of the entire back-end process is presented on figure 16. When using the Django Channels library, Daphne and Redis are required to exchange messages within the server, as mentioned in section 5.2.1. This technology will later allow for live updates of the front-end (as described in section 5.4.2).

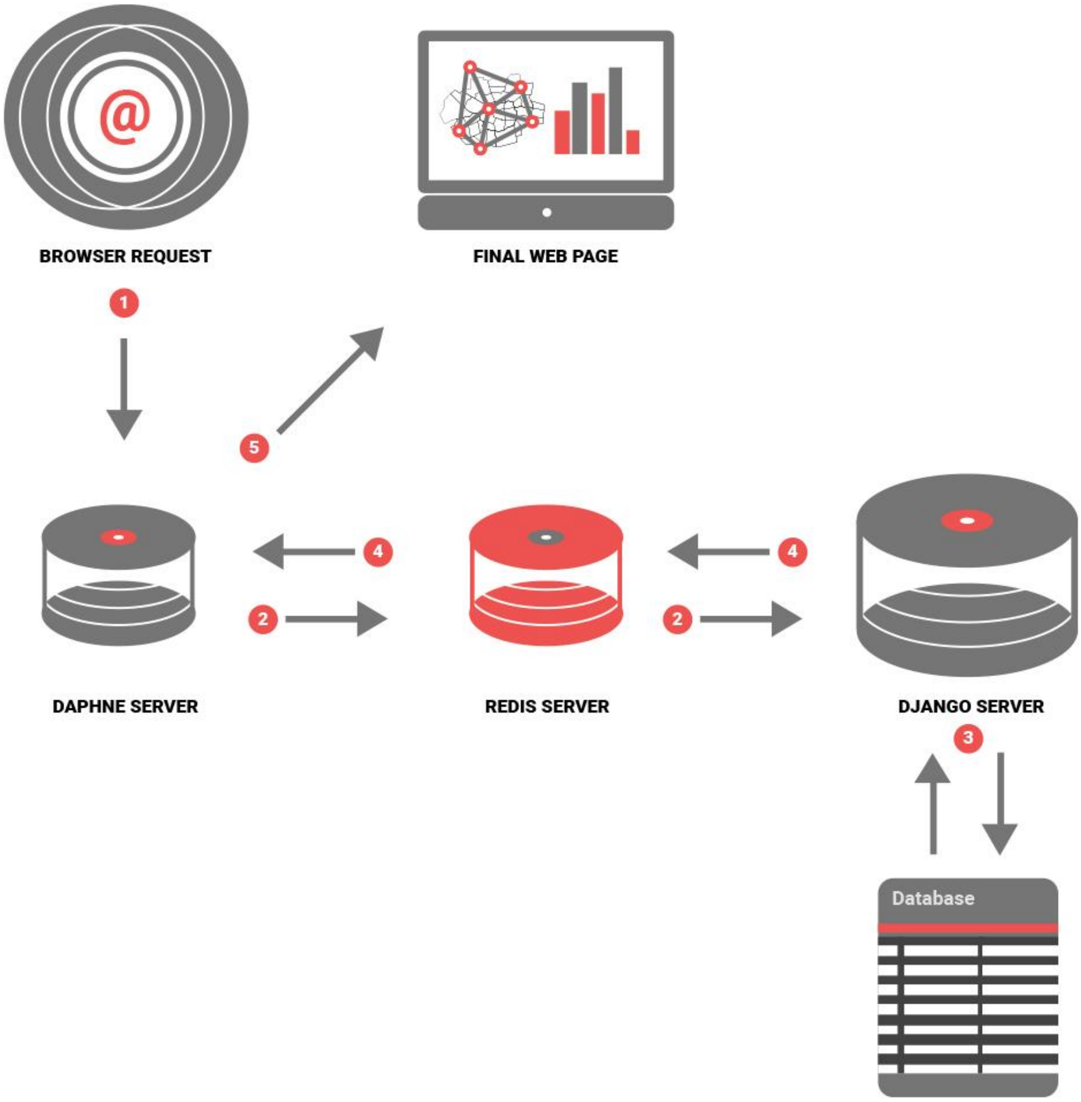


Figure 16: Overview of a request within the server

1. A request is sent from a web browser (a URL which points to our Raspberry Pi IP address is entered in the address bar)
2. Daphne server, which is listening to requests made toward our IP (on a specific port), catches up the request and pass it to our Django server application using Redis server.
3. Django retrieves the data from our database and renders the requested page (in our case there is only one page, or one 'view': the main page containing our visualization)
4. The web files (the main .html file along with all the front-end technology) are sent back to Daphne through Redis
5. Finally our Daphne server send the files to the web browser, answering the request made during step 1

5.3 Front-end Technology

As mentioned in the previous section, the application is almost ready since it can both record data and deliver it to a web browser. However the web framework, Django, can only provide certain data formatting (such as making for-loops in HTML documents). In order to perform the visualizations, additional front-end technology must be loaded with the page. This section will explain in details which technology is used and how.

5.3.1 Bootstrap

This framework consist of a CSS file a Javascript file which provides several cosmetic functionalities. The most popular and powerful one is a grid system which allows to build the sections of a web page by declaring a main container and a new column for each element. The advantage of using a technology such as Bootstrap is its compatibility across different browsers: the grid system is meant to make responsive designs, which means that it can adapt to any size of screen on different devices (e.g.: and iPad and an Android phone have different browsers, resolutions, screen sizes, …).

Bootstrap provides many other functionalities but along with the grid system this project only uses the standard font sizes and the table styling. The table are simply filled with the for-loop functionality of Django, as mentioned above.

5.3.2 Leaflet

Now that we have declared our columns, it is time to fill them in. The first visualization is the interactive map: it must display each sensors' position along with its latest measurement. Leaflet is a Javascript library which provides a complete world map with only a few lines of code.

Once we have chosen a style for the map, we can simply declare its display size and an initial location. In our case the starting point is above the city of Enschede with a slight zoom out in order to fit the sensors located on the University of Twente campus and around the city center.

The last step is to add our sensors on the map. First of all a list of the last measurements is passed on the web page and then retrieve in a Javascript array (as illustrated on figure 13). Small icons are also passed on (which represents the zone color) which will be used as a marker on the map. Leaflet provides a simple way to declare new markers, which form a small object that can be added to the map with a simple "addTo(map_name)" function. The sensor's data is therefore used to provide a latitude and longitude along with the sensor's name and its measurement. Automatically, Leaflet will add the marker at the correct location and display the name and measurement value in a pop-up when the marker is clicked.

This simplicity was the reason why this technology was chosen to display a map, it is open-source and natively works with OpenStreetMap, a free editable map of the world. The result can be seen on figure 17.

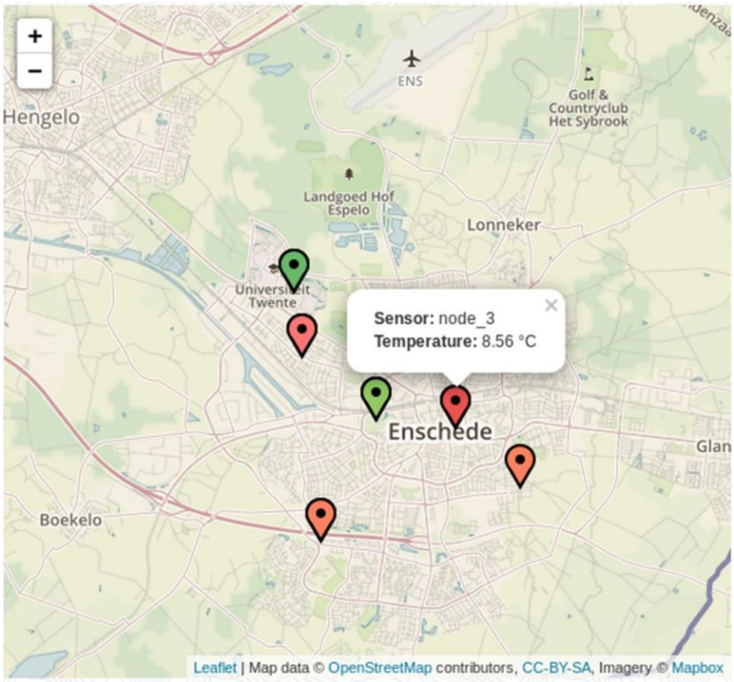


Figure 17: Map representing the sensor's location and latest measurement

5.3.3 CanvasJs

In order to build our bar and line charts, the CanvasJs library was chosen for its simplicity. In the same way Leaflet offers a visualization with only a few lines of code, CanvasJs allows to easily build very lightweight charts.

For both the line and bar charts, the data passed on by Django on page load is formatted it in a Javascript array. The bar chart is using the same object as the Leaflet map which contains the latest measurements. However the line chart is using another object which contains the last fifty measurement recorded in the database in order to provide an overview over time.

The advantage of this library is that it can automatically deal with problems such as scales differences (e.g.: the scale of the Y-axis will adapt if a new value is significantly bigger and smaller). Therefore the scale of our bar chart is automatically refreshed with new measurements. Its main disadvantage is that it is not open source and only allows free users to display charts with a link to their company at the bottom.

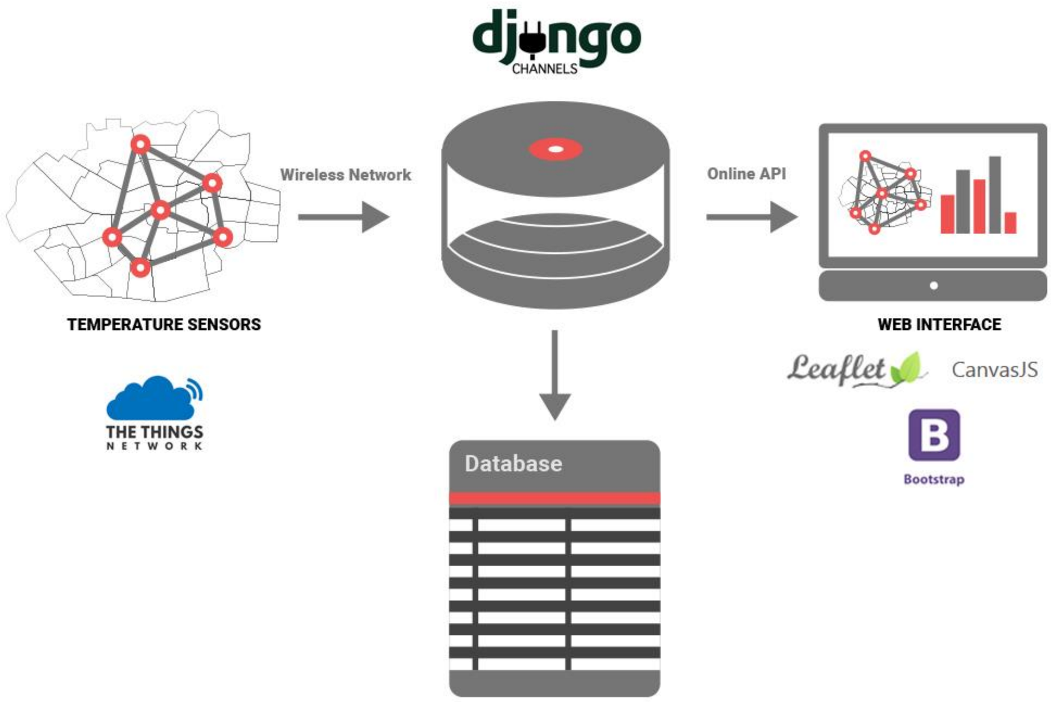


Figure 18: Final prototype technology overview

5.3.3 Additional Graphical Elements

Once the front-end technology was up and running, several graphical elements were added. First of all a legend is provided in order to clearly indicate the meaning of the color palette. This is crucial to our visualization since this choice was made even though it can be counter-intuitive, as mentioned in section 4.3.3.1.

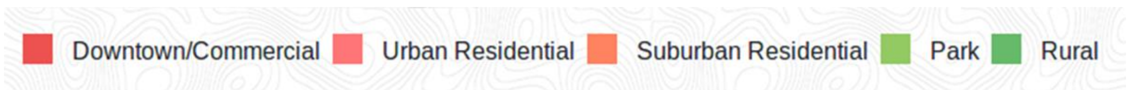


Figure 19: Legend placed under the map and bar chart on the final prototype

While the legend is helpful for the final user experience, two following elements were only added to polish the final interface and to give it a certain uniqueness. A subtle background pattern was added to give some texture to the page and a logo was made in order to make the interface easily recognizable.

Figure 20: Logo present on the final interface, including a temperature/map indicator icon

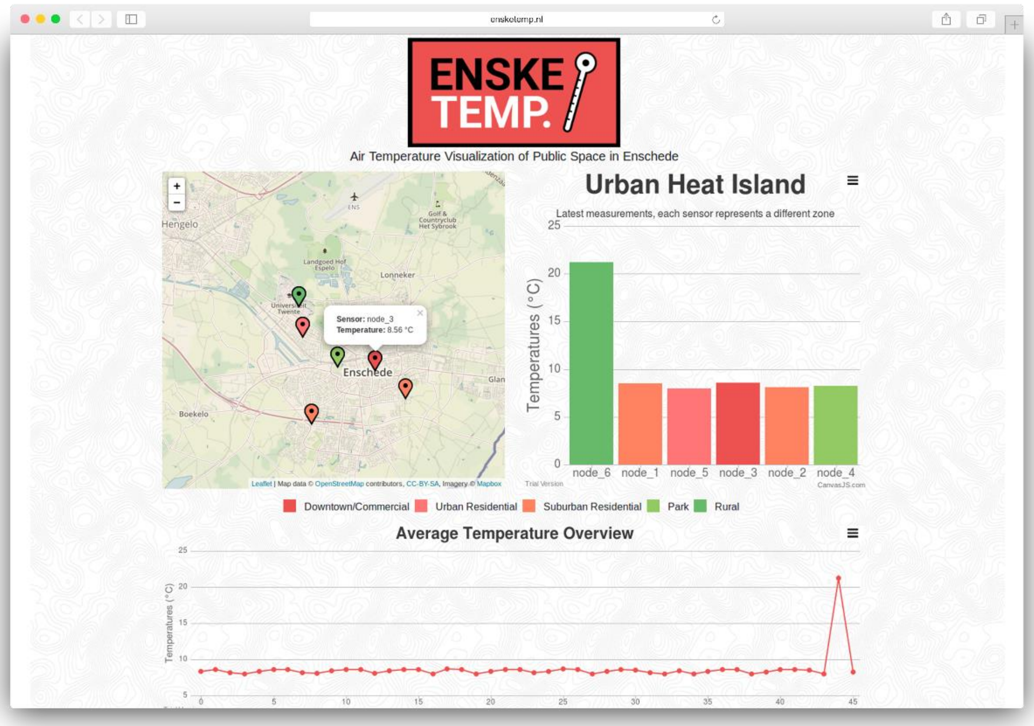


Figure 21: Mockup made using a screenshot of the front-end final prototype, a full screenshot can be seen in appendix 5

5.4 Final Prototype Software Architecture

In this final section the complete architecture of the final software prototype is described in details. In order to cover all the existing functionalities this section is divided in two sub sections. The first sub section will describe the server's behavior when a page is requested by a client while the second section will describe how the server is gathering data from TTN.

5.4.1 Web Page Request

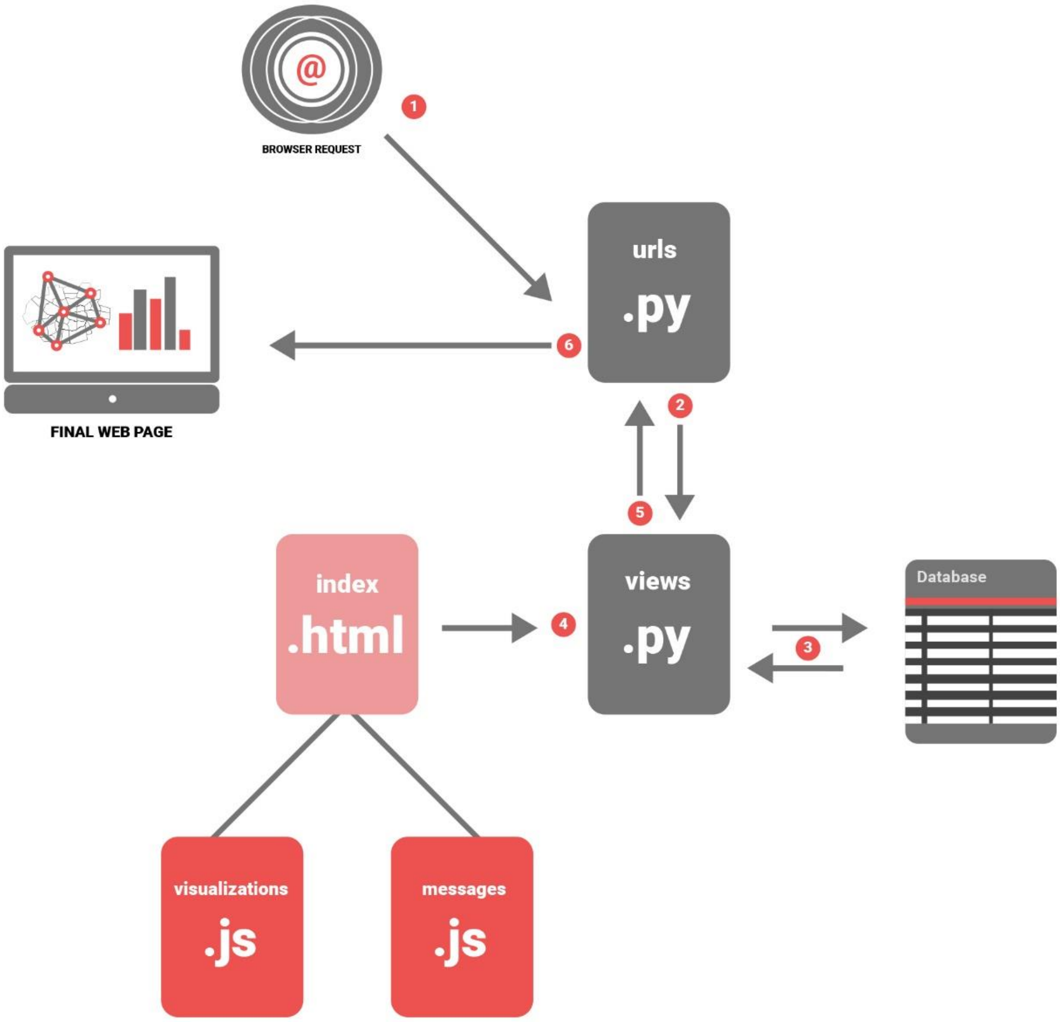


Figure 22: Software overview for a page request

The server's software behavior and architecture on page request is shown on figure 22. The entire process is labelled chronologically, and described in this section.

1. A request from the client is made to the server
2. The behavior to follow depends on the request and is defined in urls.py. In our case there is only one index page and its components are defined in views.py
3. To prepare to return the web page, views.py will make a call to the database to retrieve certain elements (e.g.: the latest measurements for each sensor node)
4. views.py also defines which HTML file to use. In our case it is index.html, which will also load all the associated files (the local Javascript files as shown above, but also CSS files and external Javascript library files)
5. The web page along with the data is then returned, as requested by urls.py

6. Finally the client's request is answered with a full web page, containing both the HTML, its CSS, the Javascript files along with data from the database.

5.4.2 Data Gathering

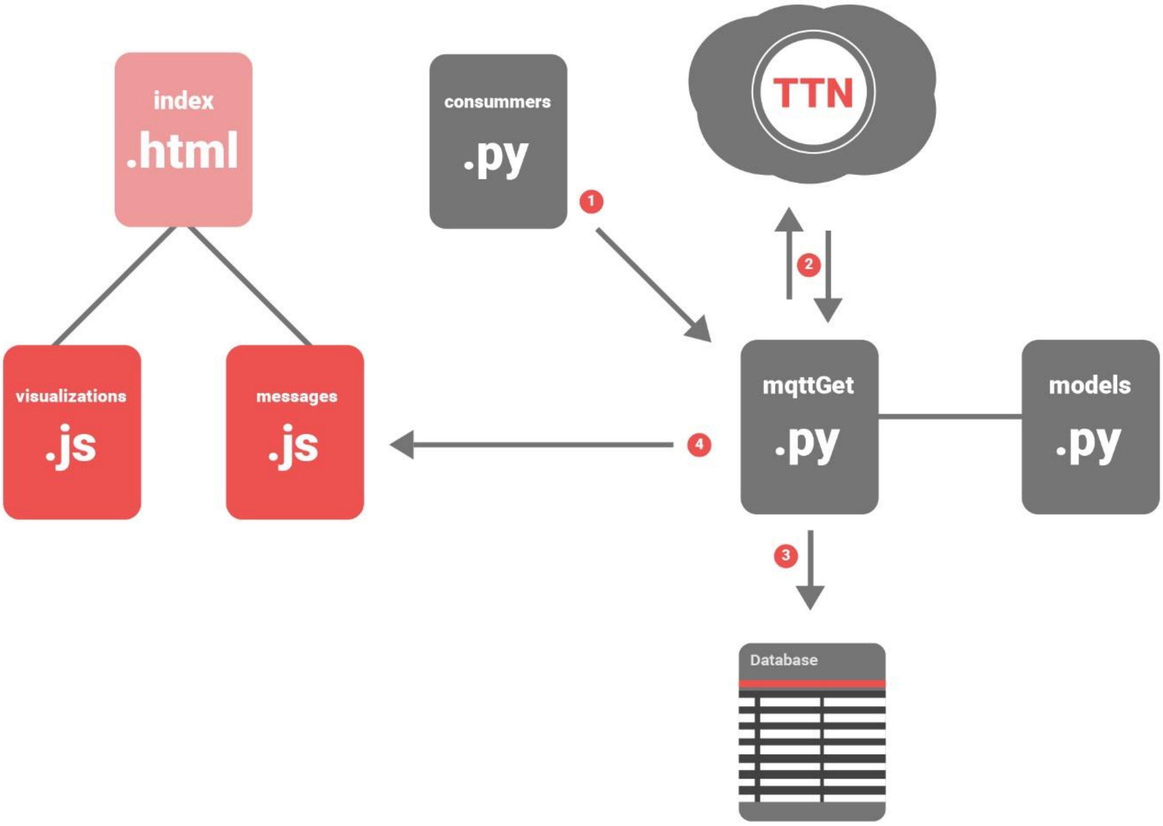


Figure 23: Software overview for data gathering

On top answering requests, our server must constantly gather data pushed by the sensors on TTN. The entire process can be seen on figure 23, and its chronological labels are described in this section.

1. On startup consummers.py will call mqttGet.py which contains the main functions to communicate with TTN.
2. The script will maintain a constant connection to the TTN application, detecting any new information pushed online by the sensors.
3. Following the database model defined in models.py, mqttGet.py will create an object which contain all the measurement information (as described in 5.2.2) and store it in the database. This happens with every detected response from TTN.
4. Finally, mqttGet.py send a message to all connected clients containing the measurement that was just received. This will naturally not happen if no clients are connected.

As we can see above the mqttGet.py file handles most of the interaction with both TTN, the database and even the client when it is connected. The full commented file can be consulted in the appendix 4.

6. Evaluation Phase

The evaluation phase will consist of two different type of tests: on one hand the performances of the server will be tested (e.g.: how fast can it deliver a web page?) and on the other hand a user based evaluation will be performed with the final interface.

6.1 Goal(s)

6.1.1 Performance evaluation

The goal of this first evaluation is to determine the efficiency of our back and front-end solution. Several aspect will be tested such as the response time in different scenarios and the compatibility of the interface with several machines.

The first test will measure the response time from a client on the internet to the Raspberry Pi in order to measure raw network performances. The second test will consist of loading our web page on different machines in order to evaluate both the full page loading duration and the compatibility of the front-end with different operating system and web browsers. The third test will determine how fast the server is updating the web page when new measurements are being pushed on The Things Network. Finally a network packet analyzer software (Wireshark), will be used to determine the overall performances while loading a page and to inspect potential issues if the previous results are not satisfying.

6.1.2 User based evaluation

Our second test will determine if the design choices made for the final prototype are clear and affordable for the final user to understand. The user based evaluation will consist of two phases: one without any indication about the nature of the project and one with background information. For each phase participants will answer a short questionnaire consisting of both multiple choices questions and open questions. Each participant is asked to read and sign a consent form before starting the test.

The result, positive or negative, will be processed in order to provide further recommendations via analyzing the motivations for such feedback.

6.2 Methods

6.2.1 Performance evaluation

6.2.1.1 Ping test

The very first performance test will consist of a simple software utility called “ping”, which is used to measure the round trip time from a host to another, using the Internet Protocol (IP).

This test will be performed from the University of Twente network and two addresses will be tested in order to compare the result: Google's public DNS server address and our Raspberry Pi's IP address. Comparing the two results will simply provide a rough estimation of the server's rough performance when accessed via the internet.

6.2.1.2 Responsiveness test

The interface webpage will be loaded on three different machines, which will allow to evaluate another aspect than pure response time: the cross platform compatibility.

The website will be loaded on a smartphone running Android using Google Chrome (in portrait mode, with a 1080x1920 resolution), on a tablet running iOS using Safari (in landscape mode, with a 1024x768 resolution) and finally on a laptop running Windows 7 using Firefox (with a 1600x900 resolution). Finally a screenshot comparison will allow for a responsiveness evaluation in order to determine if all the elements of the page are properly displayed on the devices.

6.2.1.3 Live update test

As described in the previous chapter, our web application can send live data updates to the web browser without requiring a page reload from the client. In this test two windows are setup side by side: on one hand a small Python script will be running which connects to the TTN application and prints our new measurements as soon as they are pushed on the network. On the other hand a web browser with the web page already loaded.

The performance will be evaluated by starting a timer when the information has been printed on the Python console and stopping it when the web page effectively displayed the new measurement.

6.2.1.4 Wireshark inspection

In order to obtain insights on the interaction between our client and server, Wireshark will be used while loading the web page. This will not only measure the overall performances, it will also give us a precise record of the packets exchanged between the client and the server.

This final performance test will provide additional information, which will be useful to diagnose unexpected results or to detect abnormal behaviors from both the client and the server.

6.2.2 User based evaluation

6.2.2.1 Uninformed test

During this first phase the participants will have the web page interface presented to them without any further explanations. Certain elements are hidden on the page in order to make sure that there is no information available apart from the visualizations. The page logo and title, sensors pop-up on the map, color labels, charts titles, chart axis labels and table title are hidden.

Participants will be free to interact with the page before answering a questionnaire. The questions, which can be found in appendix 6, were chosen to determine the clarity of the interface. To provide participants with enough time to explore the interface, the questionnaire is filled by the observer on a separate machine.

This first test will determine how easy it is to understand the nature of the data displayed on the page and also the level of confusion the chosen design may trigger.

6.2.2.2 Informed test

Once the first phase is finished, a similar web page will be introduced to the participants which contains all the previously hidden information. A short brief of the entire project is also given by the observer to explain the project's context. Any questions about the questions will also be answered by the observer at this point.

This second phase will focus on gathering participant's opinions on the interface, given the nature of the project. The questions will therefore focus on the appeal participants may or may not have with the final interface: are the size, colors and disposition of the elements useful or confusing? Finally an open question will allow suggestions for improvement.

In order to minimize the observer's influence on the answers, participants will be invited to fill in this part of the questionnaire themselves.

6.3 Results

6.3.1 Performance evaluation

6.3.1.1 Ping test

The 'ping' tool shipped with Windows 7 was used for this test and this section will briefly discuss one particular value: the average round trip time. As we can see on figure 24, the round trip time is longer for our server than for Google's server. The difference is rather small

(6 milliseconds) however it is noted that the average round trip time is more than doubled for our server.

This first test gives us very little information but it provides the evidence that if there is any slow response time detected in the future it will be because of the software solution implemented rather than the Raspberry Pi and the network from which it accesses the internet.

```
Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=5ms TTL=60
Reply from 8.8.8.8: bytes=32 time=5ms TTL=60
Reply from 8.8.8.8: bytes=32 time=4ms TTL=60
Reply from 8.8.8.8: bytes=32 time=4ms TTL=60

Ping statistics for 8.8.8.8:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 5ms, Average = 4ms

C:\Users\MADCATO-T420s>ping 130.89.93.97

Pinging 130.89.93.97 with 32 bytes of data:
Reply from 130.89.93.97: bytes=32 time=19ms TTL=64
Reply from 130.89.93.97: bytes=32 time=7ms TTL=64
Reply from 130.89.93.97: bytes=32 time=9ms TTL=64
Reply from 130.89.93.97: bytes=32 time=7ms TTL=64

Ping statistics for 130.89.93.97:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 7ms, Maximum = 19ms, Average = 10ms
```

Figure 24: Results of the two ping tests, with Google's DNS server IP (8.8.8.8) first and our server (130.89.93.97)

6.3.1.2 Responsiveness test

The results of this test are straightforward: the website could load without any issues on all the three platform and browsers. As mentioned in chapter 5, Bootstrap's grid system is taking care of resizing all the elements depending on the device's resolution.

Therefore the design is consistent on every device which is both a positive and slightly negative result: while performing the test it was obvious that the mobile interface is a bit too small and therefore does not allow for a perfect navigation. Screenshots were taking and integrated in a mockup on figure 25 to make it easier to directly compare the three types of display.

Figure 25: Mockup of the webpage loaded on three devices, made using screenshots

6.3.1.3 Live update test

The result from this test was quite hard to measure since the webpage almost immediately pick up the information recorded in the database. A stop watch gave a result of 580 milliseconds, but this may mostly include the time taken to actually press 'start' and 'stop'. A screenshot of the setup used to perform this test can be found on figure 26.

The result is satisfying: the delay is almost non-existent which means that the interface can display new values as soon as a sensors node pushed it on The Things Network platform.

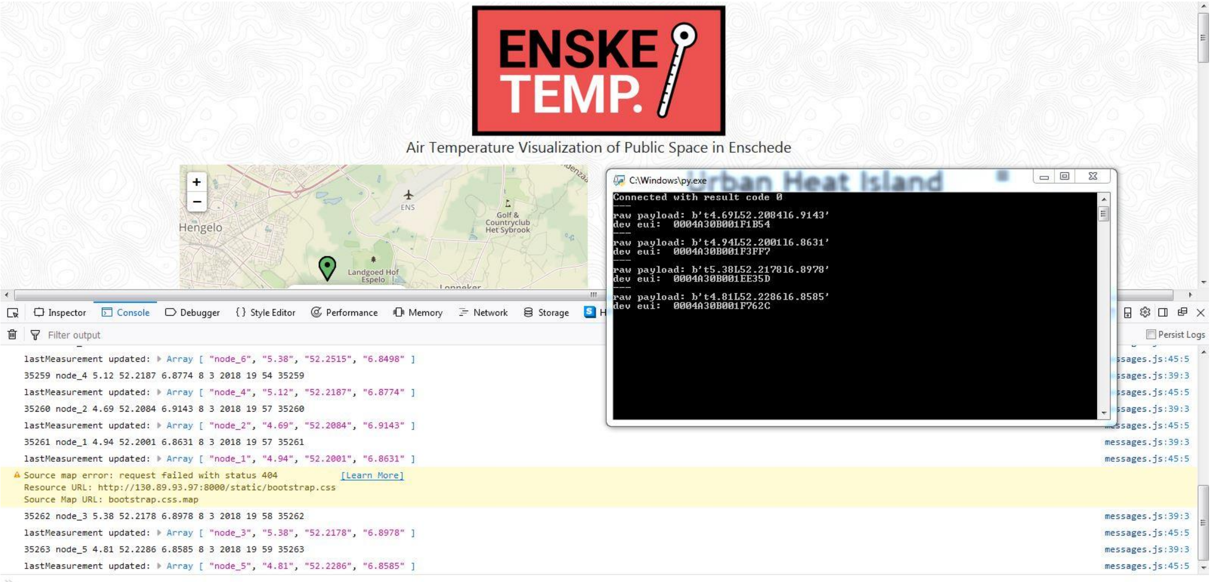


Figure 26: Python script displaying values from TTN above the web browser's console printing recevied values from the server

6.3.1.4 Wireshark inspection

This last test was the most important since it will measure the actual loading page of our web page. Overall the results are highly disappointing since our web page took almost two minutes to completely load. Such performances greatly affect the final user experience as the website appears to be timed out until the server actually send the page's information.

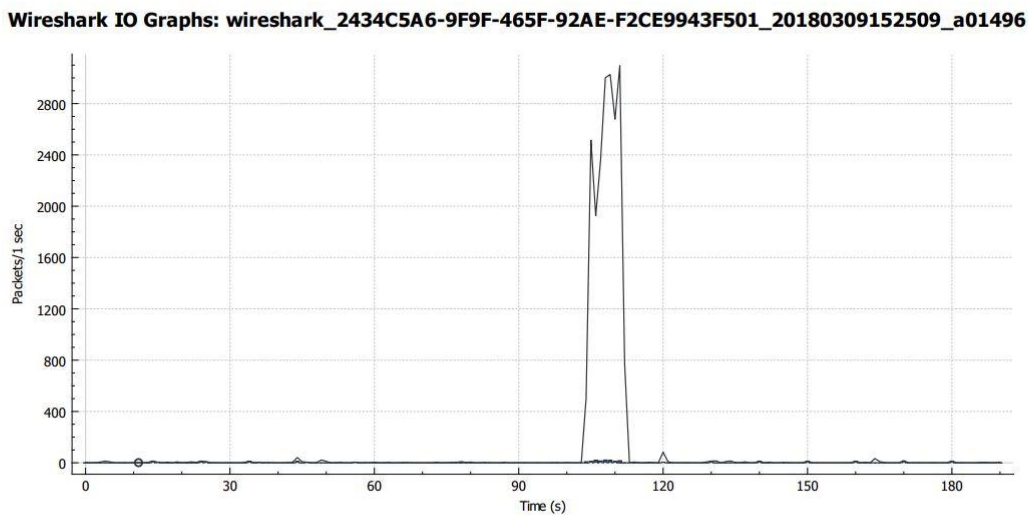
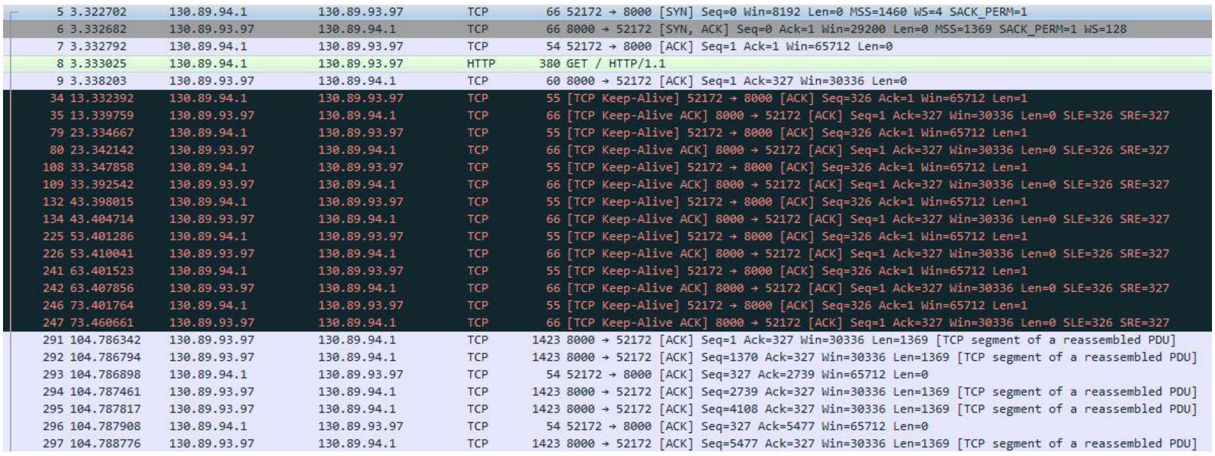


Figure 27: Wireshark IO graph showing the captured packets

As seen on figure 27, there is almost no information exchanged between the server and client for more than 90 seconds. The peak that can be seen afterwards represents the moment where our server finally sends all the information (HTML, CSS and Javascript files along with data from the database). The positive aspect is that this step does not take more than a few dozens of seconds, but there is a very long idling period before. This graph also shows small peaks after the page is loaded, which represent our WebSocket updates sent by the server.



5	3.322702	130.89.94.1	130.89.93.97	TCP	66	52172 → 8000 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
6	3.332682	130.89.93.97	130.89.94.1	TCP	66	8000 → 52172 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1369 SACK_PERM=1 WS=128
7	3.332792	130.89.94.1	130.89.93.97	TCP	54	52172 → 8000 [ACK] Seq=1 Ack=1 Win=65712 Len=0
8	3.333025	130.89.94.1	130.89.93.97	HTTP	380	GET / HTTP/1.1
9	3.338203	130.89.93.97	130.89.94.1	TCP	60	8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0
34	13.332392	130.89.94.1	130.89.93.97	TCP	55	[TCP Keep-Alive] 52172 → 8000 [ACK] Seq=326 Ack=1 Win=65712 Len=1
35	13.339759	130.89.93.97	130.89.94.1	TCP	66	[TCP Keep-Alive ACK] 8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0 SLE=326 SRE=327
79	23.334667	130.89.94.1	130.89.93.97	TCP	55	[TCP Keep-Alive] 52172 → 8000 [ACK] Seq=326 Ack=1 Win=65712 Len=1
80	23.342142	130.89.93.97	130.89.94.1	TCP	66	[TCP Keep-Alive ACK] 8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0 SLE=326 SRE=327
108	33.347858	130.89.94.1	130.89.93.97	TCP	55	[TCP Keep-Alive] 52172 → 8000 [ACK] Seq=326 Ack=1 Win=65712 Len=1
109	33.392542	130.89.93.97	130.89.94.1	TCP	66	[TCP Keep-Alive ACK] 8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0 SLE=326 SRE=327
132	43.398015	130.89.94.1	130.89.93.97	TCP	55	[TCP Keep-Alive] 52172 → 8000 [ACK] Seq=326 Ack=1 Win=65712 Len=1
134	43.404714	130.89.93.97	130.89.94.1	TCP	66	[TCP Keep-Alive ACK] 8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0 SLE=326 SRE=327
225	53.401286	130.89.94.1	130.89.93.97	TCP	55	[TCP Keep-Alive] 52172 → 8000 [ACK] Seq=326 Ack=1 Win=65712 Len=1
226	53.410041	130.89.93.97	130.89.94.1	TCP	66	[TCP Keep-Alive ACK] 8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0 SLE=326 SRE=327
241	63.401523	130.89.94.1	130.89.93.97	TCP	55	[TCP Keep-Alive] 52172 → 8000 [ACK] Seq=326 Ack=1 Win=65712 Len=1
242	63.407856	130.89.93.97	130.89.94.1	TCP	66	[TCP Keep-Alive ACK] 8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0 SLE=326 SRE=327
246	73.401764	130.89.94.1	130.89.93.97	TCP	55	[TCP Keep-Alive] 52172 → 8000 [ACK] Seq=326 Ack=1 Win=65712 Len=1
247	73.460661	130.89.93.97	130.89.94.1	TCP	66	[TCP Keep-Alive ACK] 8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=0 SLE=326 SRE=327
291	104.786342	130.89.93.97	130.89.94.1	TCP	1423	8000 → 52172 [ACK] Seq=1 Ack=327 Win=30336 Len=1369 [TCP segment of a reassembled PDU]
292	104.786794	130.89.93.97	130.89.94.1	TCP	1423	8000 → 52172 [ACK] Seq=1370 Ack=327 Win=30336 Len=1369 [TCP segment of a reassembled PDU]
293	104.786898	130.89.94.1	130.89.93.97	TCP	54	52172 → 8000 [ACK] Seq=327 Ack=2739 Win=65712 Len=0
294	104.787461	130.89.93.97	130.89.94.1	TCP	1423	8000 → 52172 [ACK] Seq=2739 Ack=327 Win=30336 Len=1369 [TCP segment of a reassembled PDU]
295	104.787817	130.89.93.97	130.89.94.1	TCP	1423	8000 → 52172 [ACK] Seq=4108 Ack=327 Win=30336 Len=1369 [TCP segment of a reassembled PDU]
296	104.787908	130.89.94.1	130.89.93.97	TCP	54	52172 → 8000 [ACK] Seq=327 Ack=5477 Win=65712 Len=0
297	104.788776	130.89.93.97	130.89.94.1	TCP	1423	8000 → 52172 [ACK] Seq=5477 Ack=327 Win=30336 Len=1369 [TCP segment of a reassembled PDU]

Figure 28: First packets sent between the client and the server

In order to see what is actually happening, Wireshark allows to see in details which packets are being exchanged between the server and the client. As we can see on figure 28 there is a rather long exchange of 'TCP Keep-alive' packets (shown with a black background). This means that while our server acknowledge the request from the client, it is taking too much time to answer therefore the client is starting this 'keep-alive' sequence in order to maintain the connection alive while waiting for an answer from the server. In other words the server is busy and cannot immediately answer the client's request.

The reason behind such a behavior is not clear, different approaches were taken in order to resolve it:

- Killing the Raspberry Pi's Ubuntu desktop environment to significantly reduce CPU usage
- Commenting our the part of the server's code which looks up for updates from TTN to remove all background processes
- Stop passing database information to the front-end in order to remove any database call

None of these approaches improved the server's reaction time.

6.3.1.5 Performance evaluation conclusion

There is one major performance issue with the implemented solution: the web page takes up to two minutes to load on a device due the timeout described in the previous section. While this a serious issue there are still some positive aspects that can be taken into account: the interface displays properly on any kind of modern devices and once the web page is loaded it provides live updates with very little delay.

Finally this performance test allows to conclude that the issue is due to the server's software implementation rather than the Raspberry Pi's network performances. There is bottleneck somewhere in the implementation which causes the delay described in the previous section. Several recommendations will be addressed in the conclusion in order to suggest a solution to this major performance issue.

6.3.2 User based evaluation

Six different people, randomly found on the campus of the University of Twente, were asked to participate in the user based evaluation. The mean age of our sample is 30 years old and the median age is about 22 years old. This sample is rather small but should be enough to have a rough estimation of the clarity and the visual appeal of the interface.

The full questionnaire along with the results can be respectively found in appendices 6 and 7. Additionally the consent form can be found in appendix 8 and well as the same form filled by the participants in appendix 9.

6.3.2.1 Uninformed test

As described earlier participants had no information about the nature of the project at this stage. The page used to perform this first test can be seen on figure 29.

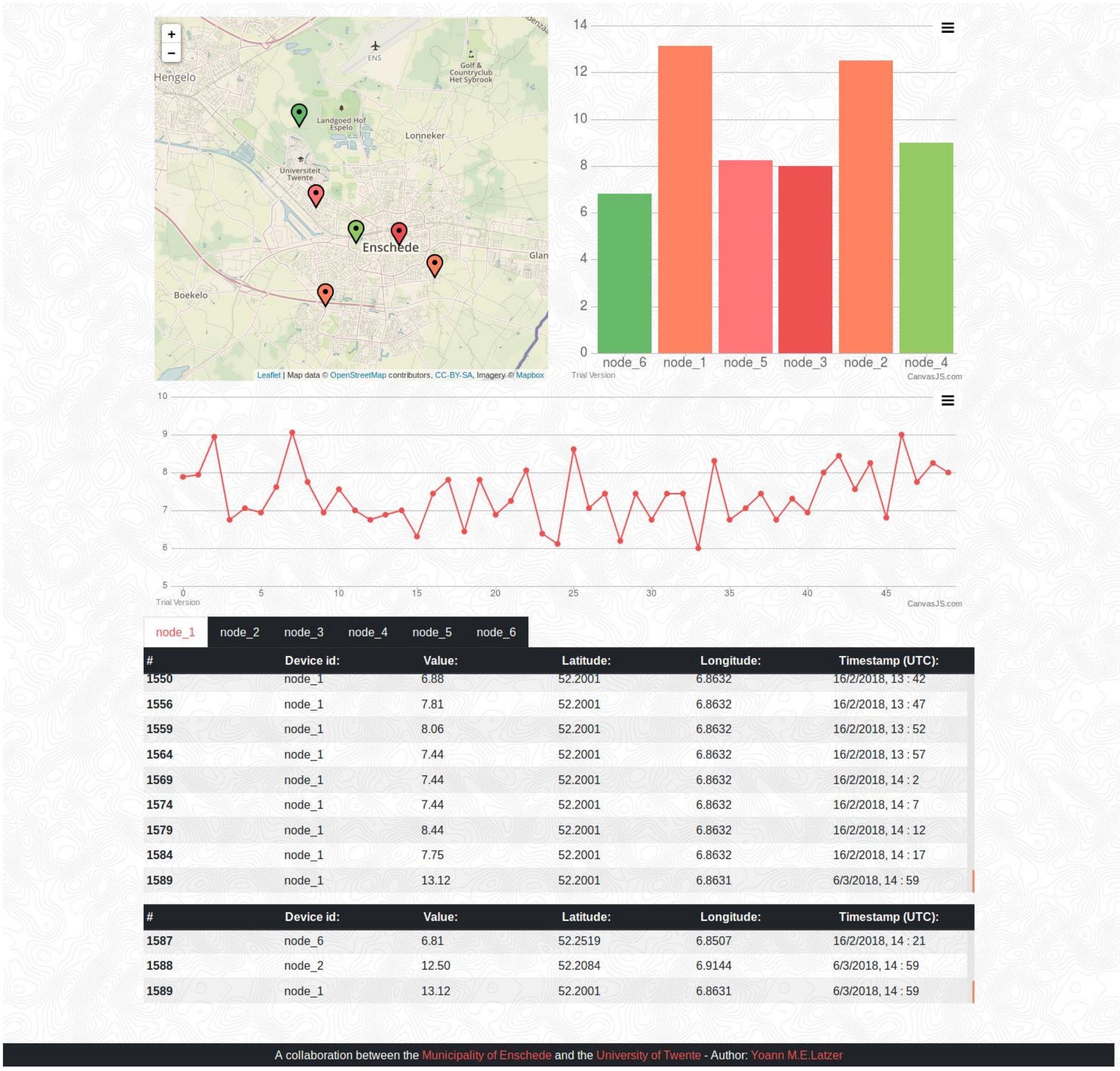


Figure 29: Page used for the uninformed test, all indications are hidden

Few participants understood without any further information that the data display was about temperature and that the color palette was based on location, as seen of figure 30 and 31. Only two out of six participants were correct, which indicates that it is rather hard to determine the nature of the visualization without any indication.

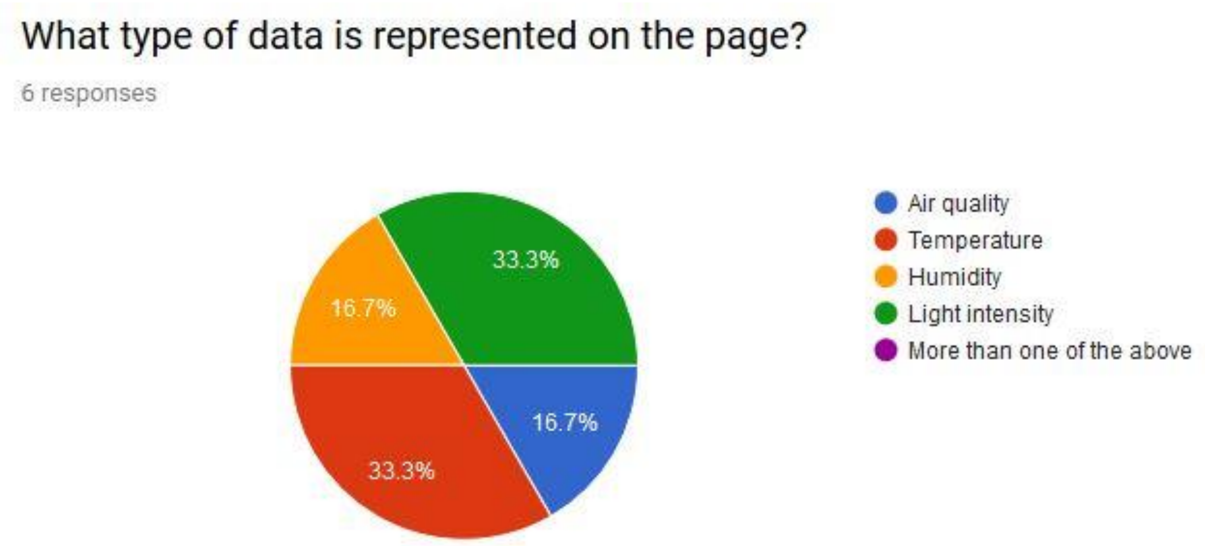


Figure 30: Results of the first question about the type of data

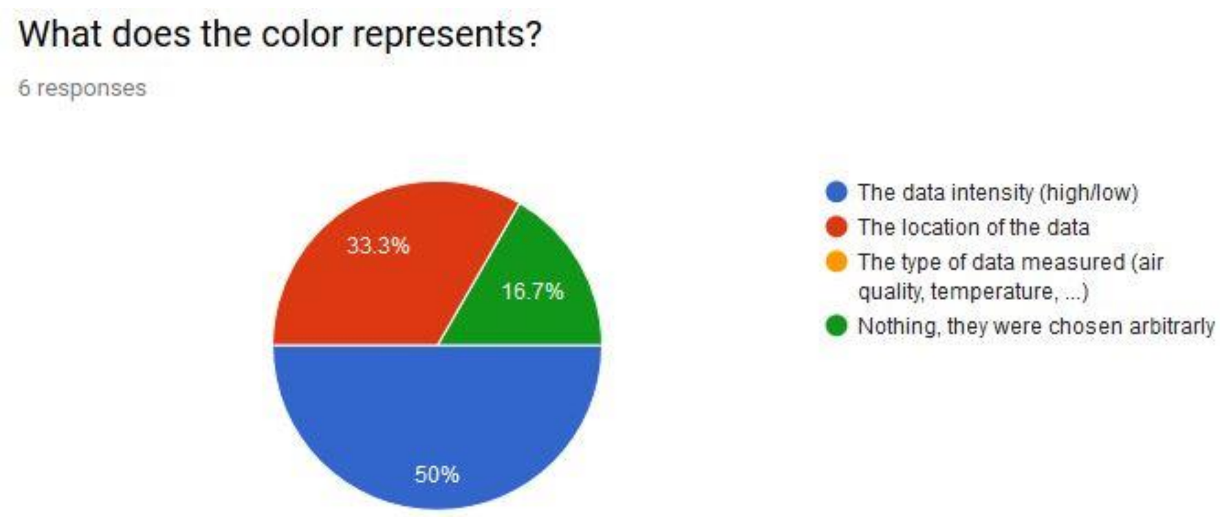


Figure 31: Results of the second question about the color palette signification

However every participant answered the two following questions correctly. These were about the relation between the map and bar chart, therefore the result indicates that the consistency between the two visualizations is working: participants were able to see that the bar chart provides information about the point on the map and that is was used to compare data from different locations.

The last question was meant to evaluate if the relation between the line chart and the rest of the visualization was clear. As seen on figure 32 more than half of the participants did understand that the data was the same, plotted over time. This indicates that the line chart's design is not clearly related to the other visualizations.

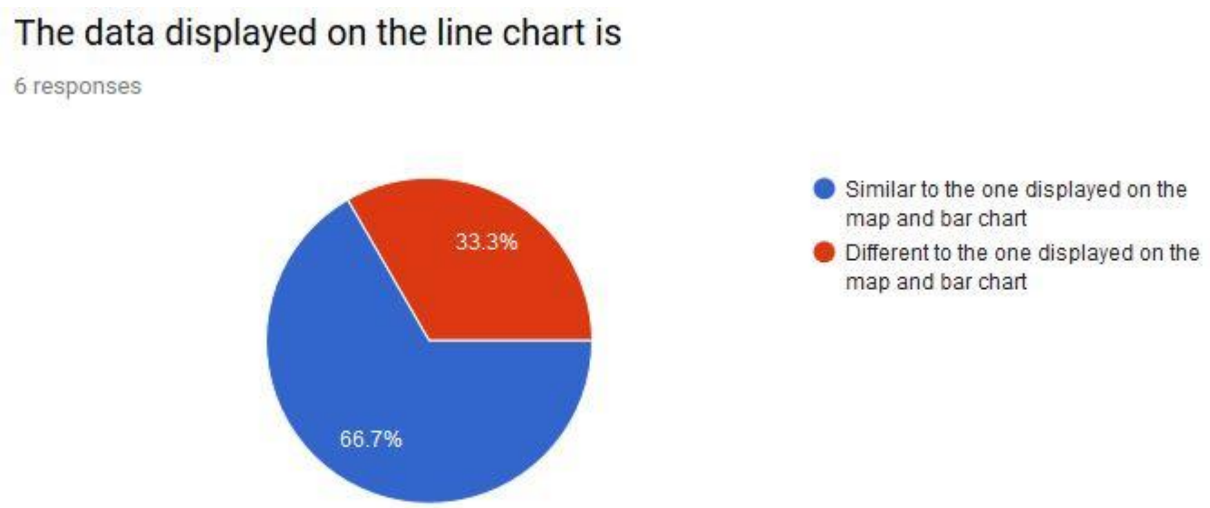


Figure 32: Results of the last question from the uninformed test, evaluating clarity of the line chart with the other visualizations

This result of the first user based evaluation phase gave mitigated results. On one hand it was difficult for the participants to understand the nature of the data but on another hand the relation between the charts appear to be rather clear. This indicates that the chosen design for the different visualizations is consistent.

6.3.2.2 Informed test

Before answering the question of this second phase, participants were provided with a complete overview of the project's nature and goal. Most of the questions are answered using a scale from one to five, one being negative and five positive. Finally an open question was used to gather further opinions and recommendations from the participants.

The first question was about the overall opinion about the design: is it appealing or not? The results are slightly positive with a mean of 3.5 which indicates that participants did not reject the design choices. No participant gave it the lowest score but none graded it with the maximum score either.

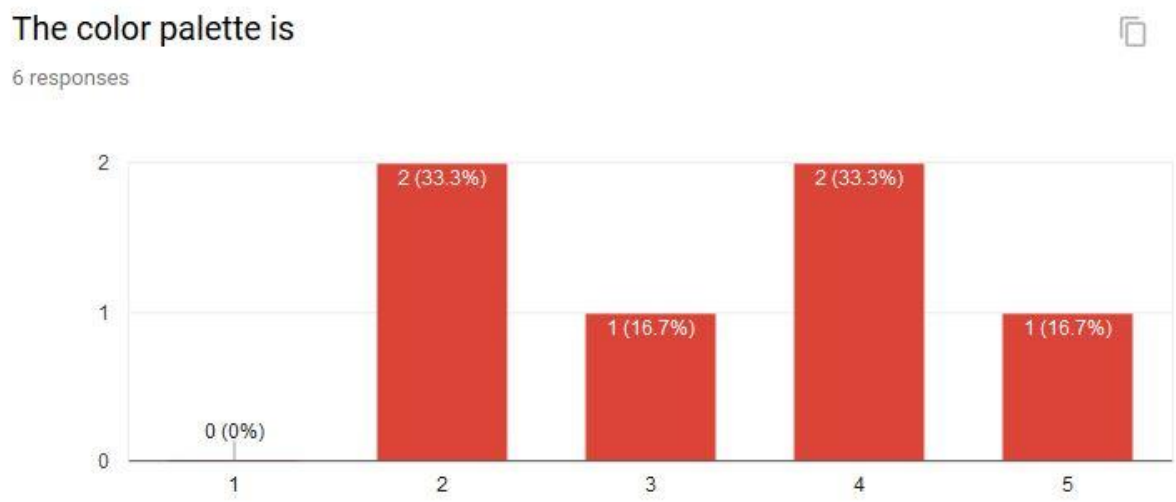


Figure 33: Result of the second question, 1 was labelled as 'confusing and 5 'useful'

The result of the following question (as seen on figure 33) about the clarity of the color palette also gave mostly positive result with a mean of 3.3 which indicated that while is it not completely confusing it could be clearer. This goes along with the result shown on figure 31 which indicates that most of the participant naturally relate color with intensity rather than location.

The most positive results were achieved with two question about the ability to get an overview of the data as well as insights about it. The mean for the first aspect was 3.8 and the mean for the second was 3.6 which indicates that all most of the participants acknowledge that the interface can be used to get both an overview and insights about the situation. However these results also indicates that there is room for improvements.

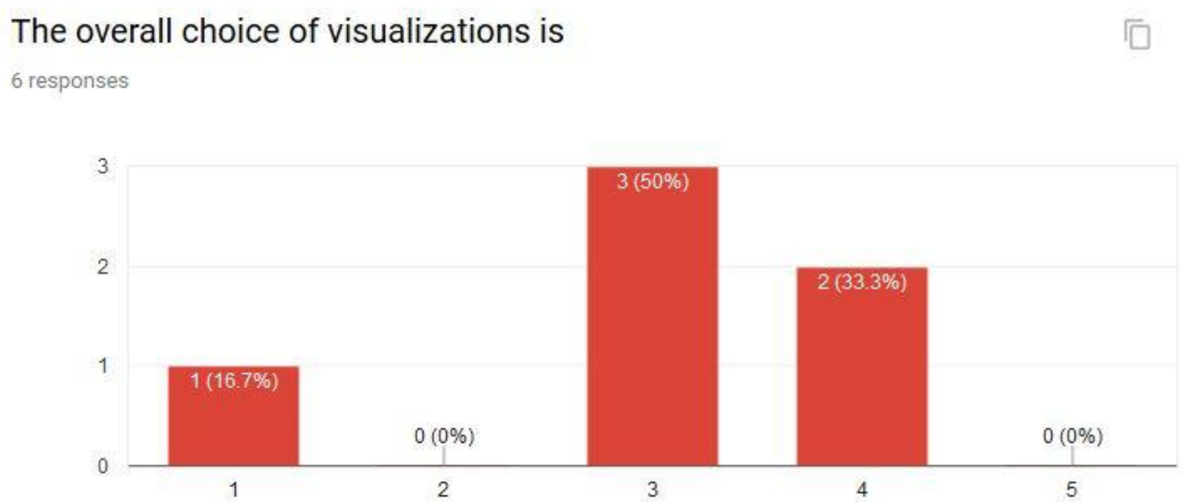


Figure 34: Result of the last question, 1 was labelled 'Confusing and 5 'Very clear'

The following question (as seen on figure 34) echoed the first one from the informed test via asking if the choices of visualization were confusing or clear. With a mean of 3 we can conclude that it is rather similar to the question about the appeal of the design. We note that one participant rated the choices of visualization as ‘Confusing’ which means again that there is room for improvement.

The final open question gave useful insights on the participants’ opinions. One comment came back from two different person and reinforced the idea that the color palette should be about temperature intensity rather than location, which confirm results discussed above. Once participant also noted that this color palette would not work for color blind people. Among other comments about the design there is one complain about the ‘bulkiness’ of the interface which indicates that it may display too much information at once.

6.3.2.3 User based evaluation conclusion

This small scale user based evaluation gave rather satisfying results overall even though it is clear that there is room for improvement. The choices made for the color palette is controversial since it did trigger several under average results and negative comments. However the choices of visualization seems to be both appealing and useful, especially when asked if it could provide both an overview and insights about the data.

7. Conclusion

After the completion of the project based on state of the art analysis, ideation and specification, a final prototype was delivered to the municipality of Enschede. It was implemented according to the description made in the realization phase before being evaluated. This final chapter will discuss the key findings and list suggestions for improvements and future developments.

7.1 Key findings

In this section the key findings will be discussed according to the research questions established in the first chapter. The completion of the project includes answering these questions as accurately as possible.

7.1.1 How to format and visualize air temperature measurements of public space in the city of Enschede for administrative and public use?

Based on the research discussed in the second chapter, a graphical representation of the gathered data is preferred to allow for fast analyzing and easy sharing of information. More precisely if is found that special based representations are particularly useful when analyzing environmental data. Hence the choice of a map in our final prototype to easily understand that this project relies on physical sensors scattered across the city of Enschede.

Furthermore the ideation phase revealed that simple, clearly separated, charts were the most performant way to provide both information overtime (line chart with temperature on the Y-axis and time on the X-axis) and a clear comparison of the latest measurements to analyze the urban heat island effect level (bar chart with temperature on the Y-axis and nodes on the X-axis). This choices were again made according to the findings discussed in the state of the art since such representation also allows for easy understanding and fast spreading of the information.

Finally simple tables containing raw data were added in order to allow for a detailed investigation of the data gathered over time.

7.1.2 What are the possible use cases of air temperature measurements for the identified stakeholders?

According to the interaction performed with the municipality of Enschede's employees along the project, it was clear that the concrete application of the final prototype would be rather experimental. There was no mention of a public release of the interface, at least on the short term. Therefore the main user case was to monitor the current urban heat island effect level via comparing relevant parts of the city. The final visualizations can be used to raise awareness about the potential of such a solution for data driven decision making. This was the main motivation for including a simple export function to the charts in order to make them easy to share among the different municipality's services.

7.1.3 How to provide an overview and useful insights of the measured air temperatures to the different stakeholders?

The use of simple visualization updated in real time seems to be the most performant to provide a realistic overview of the situation as well as insights. However the line chart was added in order to also include a clear overview of the situation in the last hours. Concrete solutions to allow for flexible (e.g.: choosing freely the time constrain of the visualization) were not implemented due to technical issues that lied outside of the scope of this project. The following section will discuss possible improvement to fully answer this question.

7.2 Further Recommendations

This section will list all the different recommendation that could be made after reflecting on the results of this project.

7.2.1 Technical recommendations

First of all the previous chapter underlined a major issue with unsatisfactory loading times. There are many different approaches to figure out exactly what is wrong with the implemented solution and the preferred one would be to deploy the Django server on an online platform in order to confirm that the problem is purely related with the way the code is written and not due to a wrong network configuration (e.g.: the University of Twente's network may not allow certain behavior, which causes the slow down).

Furthermore an API approach to collect the information from the database would both improve the performances and the scalability (e.g.: getting the latest measurements from a URL such as 'ensketemp.nl/last50measurements' instead of passing them to the front view every time the index is server to a client). This would also allow for external application to connect with the server in order to perform other types of visualization for example.

7.2.2 Visualization recommendations

In chapter 6 it was confirmed that the color palette used on the final prototype was controversial, which lead to two potential different solutions: either entirely change it to reflect the temperature level or replace the warm colors used for urban zones by gray tones to emphasis even more on the green (parks, rural) versus urban (city center, urban residential, sub-urban residential) locations.

In the end the visualizations proposed with the final prototype were rather simplistic due to the fact that most of the time was spent on setting up a full web framework and deploying it on the selected hardware. More sophisticated visualizations can be envisaged such as a 3D map representing the urban heat island effect as mentioned in the state of the art.

Additionally the data gathered with the web server could be used to make more 'traditional' communication material such as posters which could inform citizens about the best place to go in Enschede to avoid the urban heat island effect. An example of such a poster was made during the visual idea generation phase and can be seen in appendix 10.

7.2.3 City-wide temperature projection

A final recommendation concern a method that was briefly explored in order to build a city-wide temperature projection. RIVM (Rijksinstituut voor Volksgezondheid en Milieu) created a map representing the expected heat level in the Netherlands. It is rather precise and can be used along with sensor measurements to show the urban heat island effect level in the entire city. A simple example would be to take measurements in a park and project it to all the parks in the city, according to the expected heat level variation represented on RIVM's map. This could effectively provide a complete overview of the urban heat island effect in Enschede using a limited amount of sensors.

[1] a Zanella, N. Bui, a Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, 2014.

[2] Koninklijk Nederlands Meteorologisch Instituut, "Twenthe, langjarige gemiddelden, tijdvak 1981-2010," 2010.

[3] H. L. Macintyre *et al.*, "Assessing urban population vulnerability and environmental risks across an urban area during heatwaves – Implications for health protection," *Sci. Total Environ.*, vol. 610–611, pp. 678–690, 2018.

[4] W. D. Solecki *et al.*, "Mitigation of the heat island effect in urban New Jersey," *Environ. Hazards*, vol. 6, no. 1, pp. 39–49, 2005.

[5] W. V. Siricharoen and N. Siricharoen, "Infographic Utility in Accelerating Better Health Communication," *Mob. Networks Appl.*, 2017.

[6] W. Lu, "An Interactive Web Mapping Visualization of Urban Air Quality Monitoring Data of China," 2017.

[7] W. V Siricharoen and P. Cong, "Question matrix method according to divided dimensions of infographics evaluation," *Pers. Ubiquitous Comput.*, vol. 21, no. 2, pp. 219–233, 2017.

[8] J. Hjort, J. Suomi, and J. Käyhkö, "Extreme urban–rural temperatures in the coastal city of Turku, Finland: Quantification and visualization based on a generalized additive model," *Sci. Total Environ.*, vol. 569–570, no. June, pp. 507–517, 2016.

[9] B. Shao, M. Zhang, Q. Mi, and N. Xiang, "Prediction and visualization for urban heat island simulation," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6758, pp. 1–11, 2011.

[10] B. F. Liu *et al.*, "Is a picture worth a thousand words? The effects of maps and warning messages on how publics respond to disaster information," *Public Relat. Rev.*, vol. 43, no. 3, pp. 493–506, 2017.

[11] S. P. Mohanty, U. Choppali, and E. Kougianos, "Everything you wanted to know about smart cities: The Internet of things is the backbone," *IEEE Consum. Electron. Mag.*, vol. 5, no. 3, pp. 60–70, 2016.

[12] S. P. Mohanty, "Everything You Wanted to Know About Smart Cities," *IEEE Consum. Electron. Mag.*, vol. 5, no. 3, pp. 60–70, 2016.

[13] I. Beretta, "The social effects of eco-innovations in Italian smart cities," *Cities*, vol. 72, no. July 2017, pp. 115–121, 2018.

[14] D. Añón Higón, R. Gholami, and F. Shirazi, "ICT and environmental sustainability: A global perspective," *Telemat. Informatics*, vol. 34, no. 4, pp. 85–95, 2017.

[15] Libelium.com. (2010). Detecting Forest Fires using Wireless Sensor Networks with Waspmote. [online] Available at: http://www.libelium.com/wireless_sensor_networks_to_detec_forest_fires/

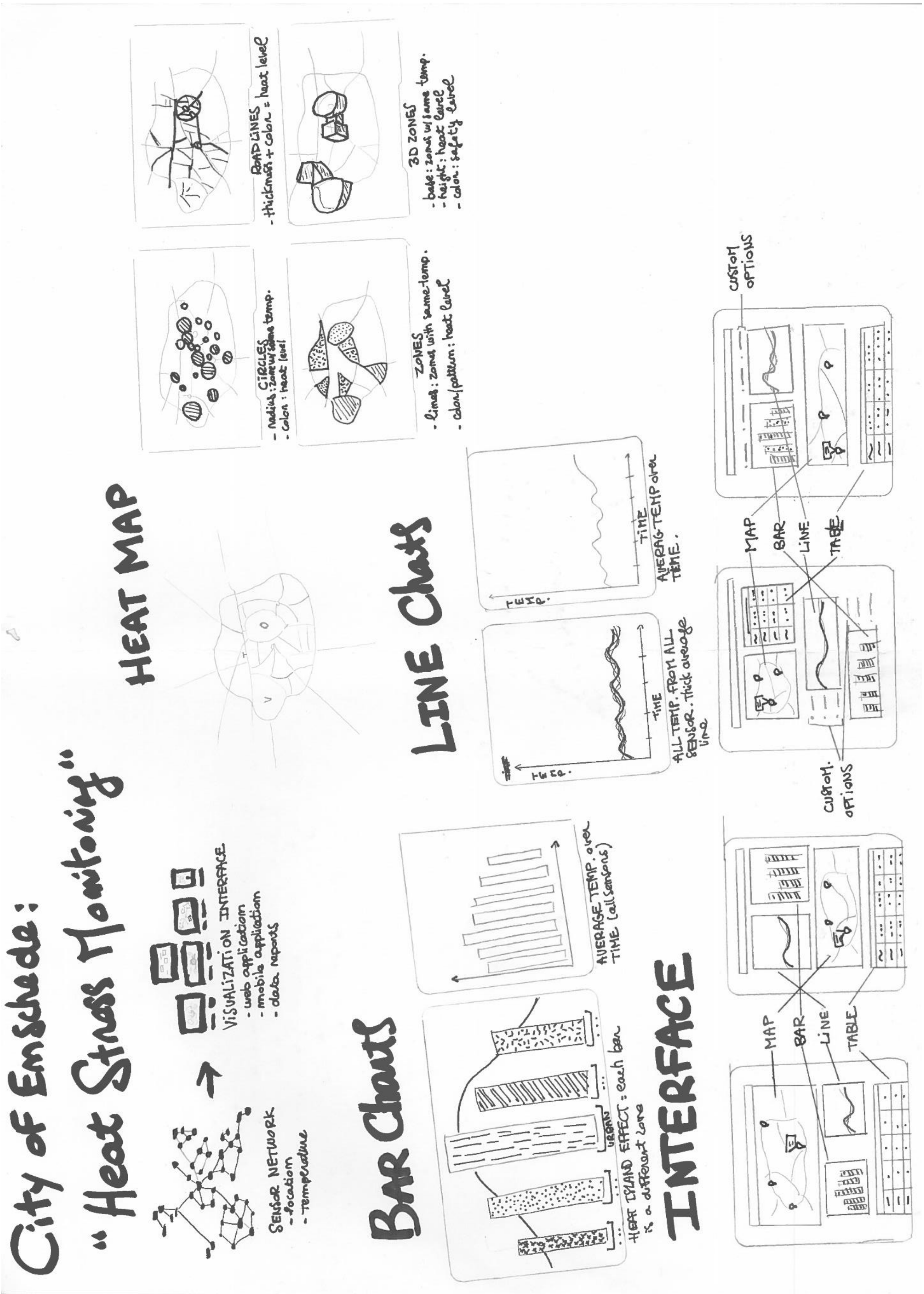
[16] Libelium.com. (2012). Libelium - Smart City project in Serbia to monitor Environmental Parameters by Public Transportation with Waspmote | Libelium. [online] Available at: http://www.libelium.com/smart_city_environmental_parameters_public_transportation_waspmote/

[17] UChicago News. (2016). Chicago becomes first city to launch Array of Things. [online] Available at: https://news.uchicago.edu/article/2016/08/29/chicago-becomes-first-city-launch-array-things

Appendices

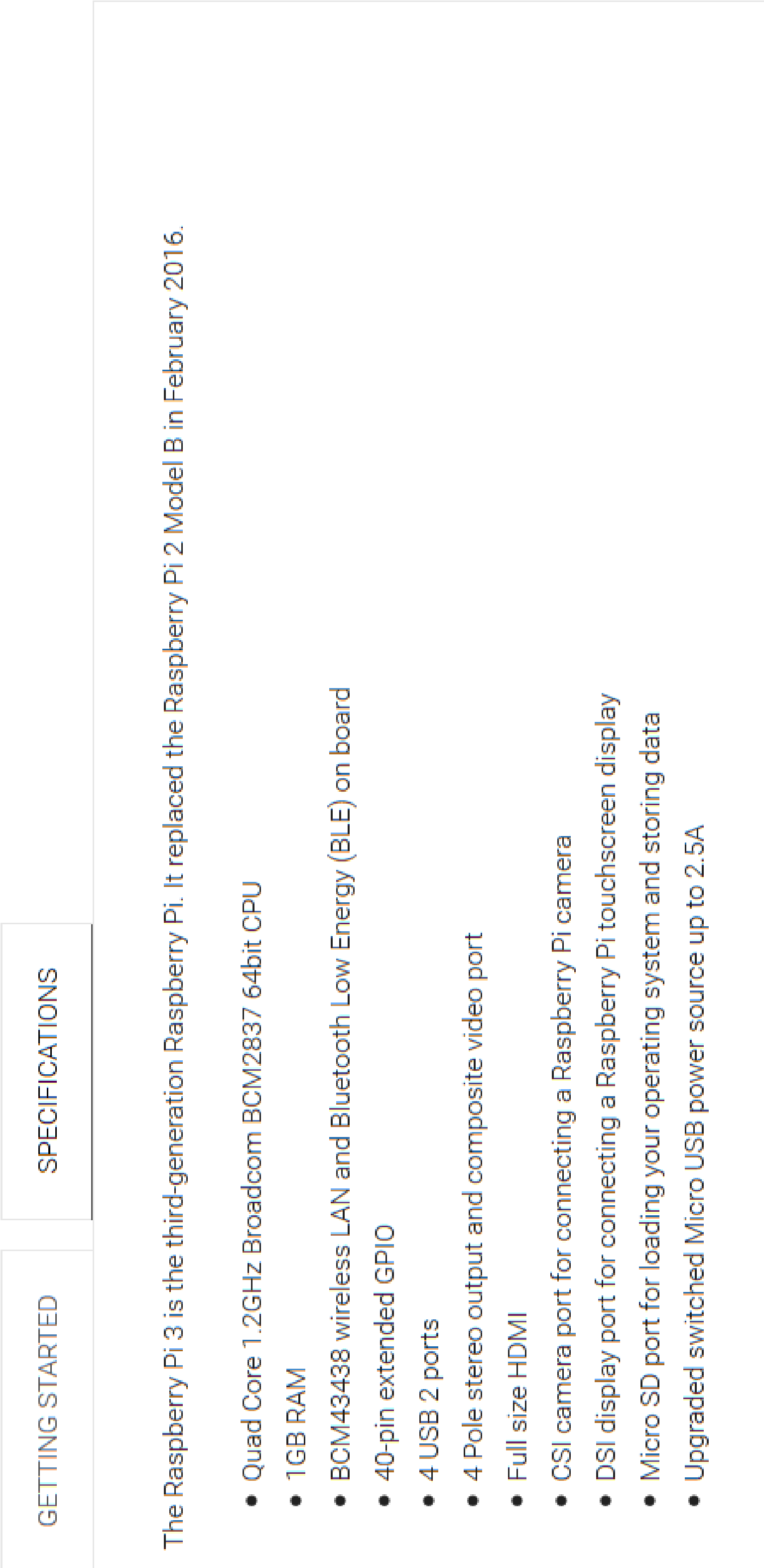
Appendix 1. Physical sensors built by Tom Onderwater (final prototype version)

Appendix 2. Selected result from the visual ideas generation phase



Appendix 3. Raspberry Pi 3 model B specifications

(Screenshot taken from raspberrypi.org/products/raspberry-pi-3-model-b/)



GETTING STARTED | SPECIFICATIONS

The Raspberry Pi 3 is the third-generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in February 2016.

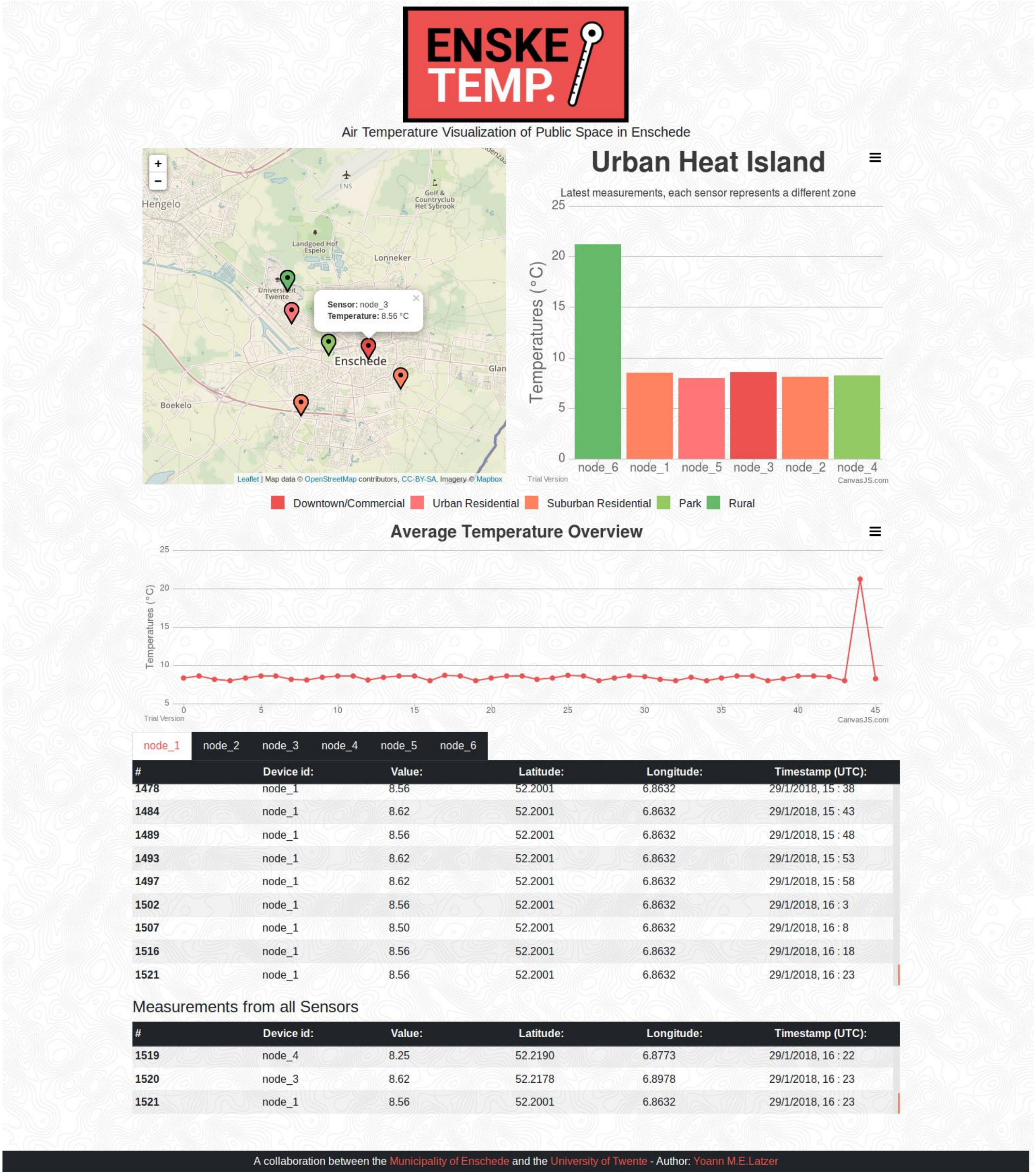
- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RAM
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
- 40-pin extended GPIO
- 4 USB 2 ports
- 4 Pole stereo output and composite video port
- Full size HDMI
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source up to 2.5A

Appendix 4. mqttGet.py, the main data gathering script

```python
# Module required: paho-mqtt (pip install paho-mqtt)
import paho.mqtt.client as mqtt
import json
import base64
import datetime
import re
from channels import Group
from .models import Measurement

# Boolean variable to prevent multiple instances of the consumer
loopStarted = False

# TTN identification variables
APPEUI = '(not shown for security reasons)'
APPID  = 'enschede_temp'
ACCESSKEY = '(not shown for security reasons)'

# Print the connection status
def on_connect(client, userdata, flag, rc):
    print("Connected with result code "+str(rc))
    client.subscribe('+/devices/+/up'.format(APPEUI))

# Run when a new measurement is received from TTN
# The json data is split into several variables before being recorded in the database according to 'Measurement' in models.py
def on_message(client, userdata, msg):
    # Decode the payload into utf-8
    j_msg = json.loads(msg.payload.decode('utf-8'))
    dev_id = j_msg['dev_id']
    # Make a time array (only to be printed since it is sometimes missing from the sensor)
    time = j_msg['metadata']['gateways'][0]['time']
    time_array = re.split('-|T|:', time)
    # Decode the binary string and store it in an array using utf-8
    payload = base64.b64decode(j_msg['payload_raw'])
    info = payload.decode("utf-8")
    info_array = re.split('t|L|l',info)

    # Print received data in Django's console
    print('--------------------------------------------------')
    print('--------------------------------------------------')
    print('metadata: ', j_msg)
    print('--------------------------------------------------')
    print('time_array: ', time_array)
    print('info_array: ', info_array)
    print('device_id: ', dev_id)

    # Store the curent time
    i = datetime.datetime.now()
    # Store the received data in a temporary variable before saving it
    temp = Measurement(measurement_type='temperature', device_id= dev_id, value=info_array[1], latitude=info_array[2], longitude=info_array[3], day=i.day, month=i.month, year=i.year, hour=i.hour, minute=i.minute)
    temp.save()

    # Print recorded data in Django's console
    print('--------------------------------------------------')
    print('Recorded in the database: ', temp)
    print('--------------------------------------------------')

    # Make a temporary object to send to the front-end
    temp_string = str(temp.id) + ' '+ str(temp)
    # Send a message to the 'sensor_data' group to be received by the front-end
    Group("sensor_data").send({'text' : temp_string})
```

```python

# Run the connection functions from paho-mqtt to connect to TTN
client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.username_pw_set(APPID, ACCESSKEY)
client.connect("eu.thethings.network", 1883, 60)
```

Appendix 5. Full screenshot of the final web page design



#	Device id:	Value:	Latitude:	Longitude:	Timestamp (UTC):
1478	node_1	8.56	52.2001	6.8632	29/1/2018, 15 : 38
1484	node_1	8.62	52.2001	6.8632	29/1/2018, 15 : 43
1489	node_1	8.56	52.2001	6.8632	29/1/2018, 15 : 48
1493	node_1	8.62	52.2001	6.8632	29/1/2018, 15 : 53
1497	node_1	8.62	52.2001	6.8632	29/1/2018, 15 : 58
1502	node_1	8.56	52.2001	6.8632	29/1/2018, 16 : 3
1507	node_1	8.50	52.2001	6.8632	29/1/2018, 16 : 8
1516	node_1	8.56	52.2001	6.8632	29/1/2018, 16 : 18
1521	node_1	8.56	52.2001	6.8632	29/1/2018, 16 : 23

Measurements from all Sensors

#	Device id:	Value:	Latitude:	Longitude:	Timestamp (UTC):
1519	node_4	8.25	52.2190	6.8773	29/1/2018, 16 : 22
1520	node_3	8.62	52.2178	6.8978	29/1/2018, 16 : 23
1521	node_1	8.56	52.2001	6.8632	29/1/2018, 16 : 23

A collaboration between the Municipality of Enschede and the University of Twente - Author: Yoann M.E.Latzer

Appendix 6. Front-end user based evaluation questionnaire

Ensketemp Front-End Evaluation

*Required

First part: uninformed test

Filled by the observer

Participant's age

Your answer

What type of data is represented on the page?

- ○ Air quality
- ○ Temperature
- ○ Humidity
- ○ Light intensity
- ○ More than one of the above

What does the color represents?

- ○ The data intensity (high/low)
- ○ The location of the data
- ○ The type of data measured (air quality, temperature, ...)
- ○ Nothing, they were chosen arbitrarly

The bar chart provides information about the elements on the map

- ◯ Yes
- ◯ No

The bar chart is used to

- ◯ Compare different types of data
- ◯ Compare data from different location type
- ◯ Compare data over time

The data displayed on the line chart is

- ◯ Similar to the one displayed on the map and bar chart
- ◯ Different to the one displayed on the map and bar chart

Second part: informed test

Filled by the participants

The overall design of the page is *

	1	2	3	4	5	
Unappealing	◯	◯	◯	◯	◯	Appealing

The color palette is *

	1	2	3	4	5	
Confusing	◯	◯	◯	◯	◯	Useful

The interface helps me getting an overview of the data *

	1	2	3	4	5	
Not at all	○	○	○	○	○	Absolutely

The interface helps me getting insights about the data *

	1	2	3	4	5	
Not at all	○	○	○	○	○	Absolutely

The overall choice of visualizations is *

	1	2	3	4	5	
Confusing	○	○	○	○	○	Very clear

What would you change about the interface? (add, remove or edit elements) *

Your answer

SUBMIT

Never submit passwords through Google Forms.

Appendix 7. Front-end user based evaluation results

QUESTIONS RESPONSES 6

6 responses

SUMMARY INDIVIDUAL

Accepting responses

First part: uninformed test

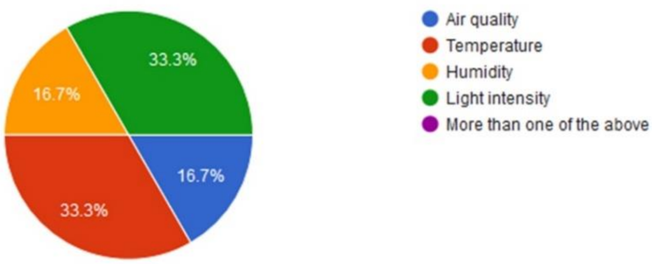
Participant's age

6 responses

22
23
20
31
65
21

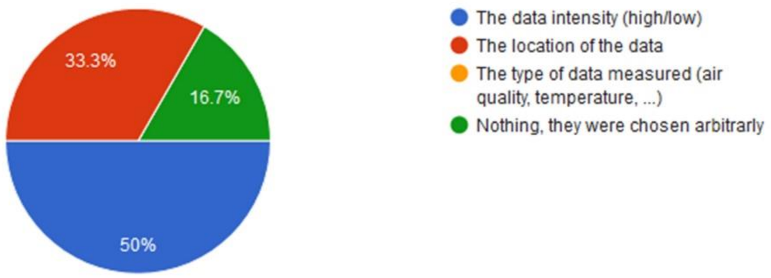
What type of data is represented on the page?

6 responses



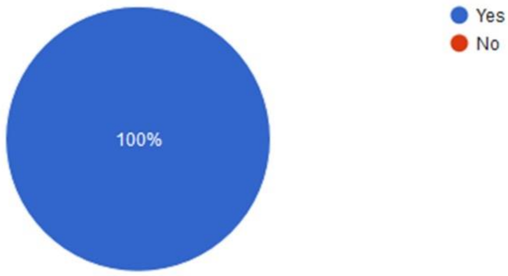
What does the color represents?

6 responses



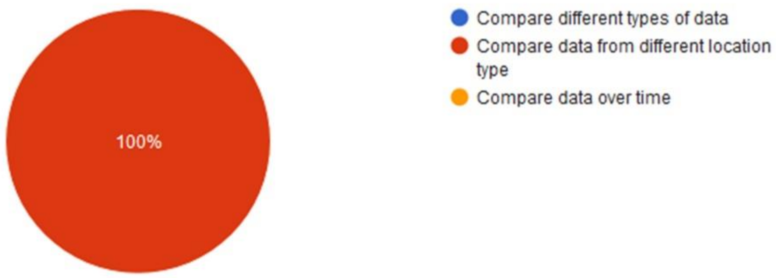
The bar chart provides information about the elements on the map

6 responses



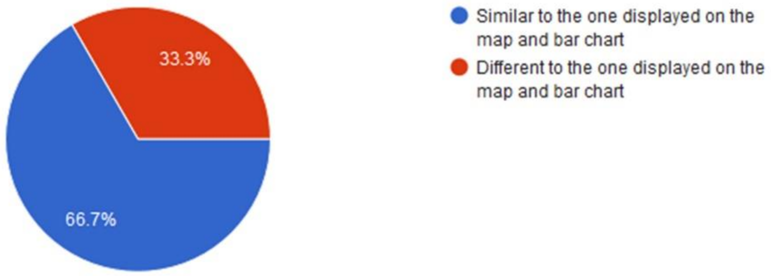
The bar chart is used to

6 responses



The data displayed on the line chart is

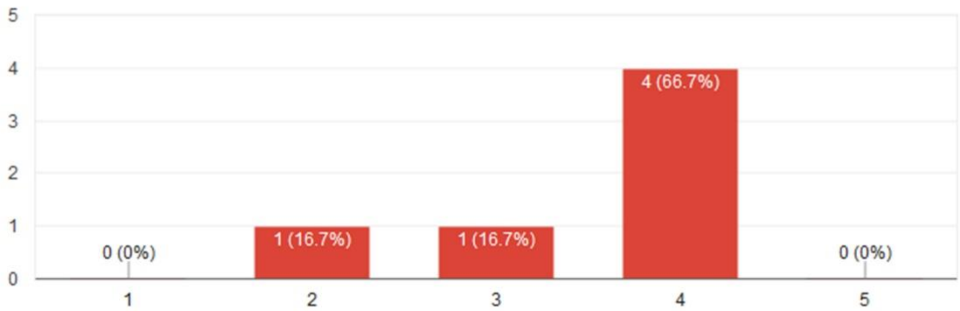
6 responses



Second part: informed test

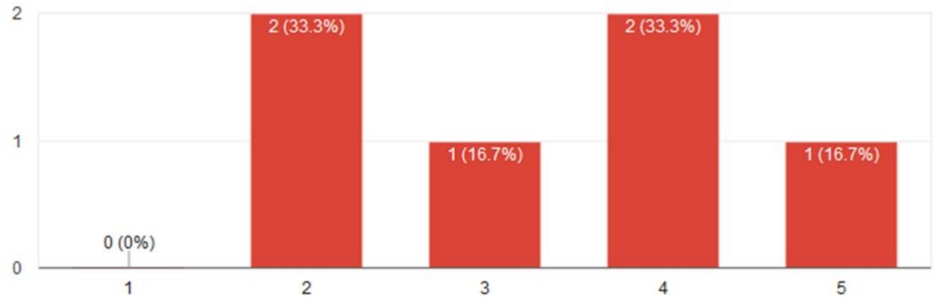
The overall design of the page is

6 responses



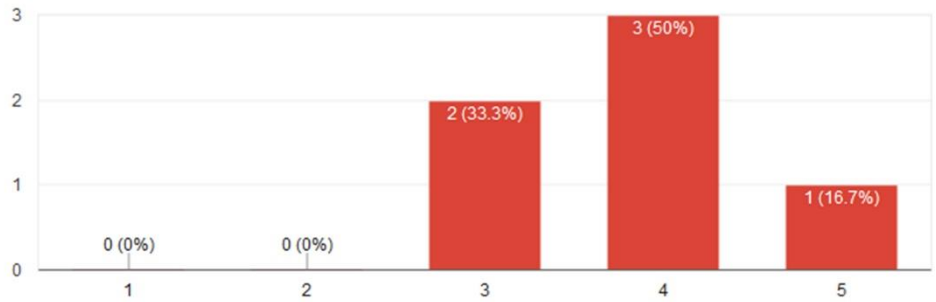
The color palette is

6 responses



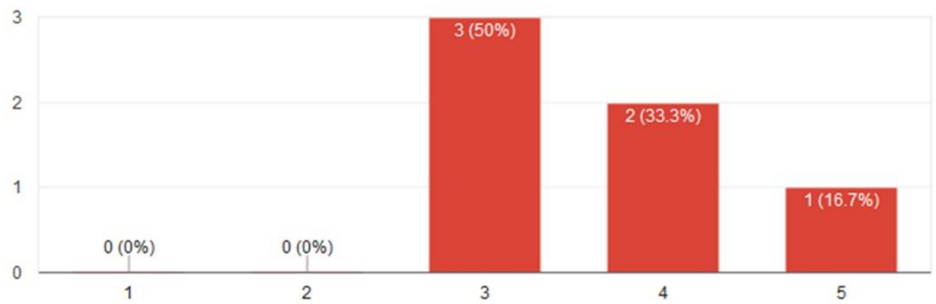
The interface helps me getting an overview of the data

6 responses



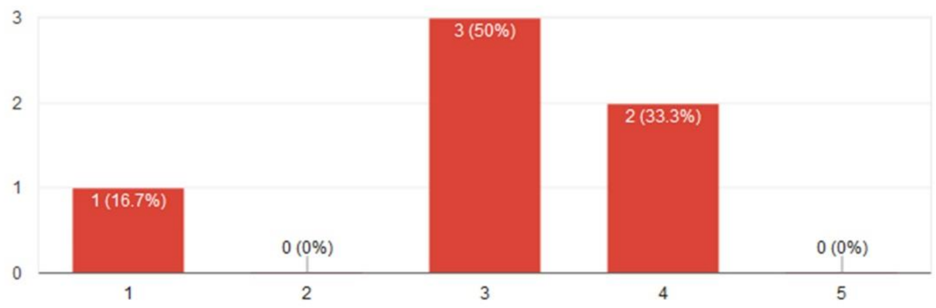
The interface helps me getting insights about the data

6 responses



The overall choice of visualizations is

6 responses



What would you change about the interface? (add, remove or edit elements)

6 responses

- use blue instead of green because i associate that more with temperature
- at first I thought the color to be reperesentative of temprature, only with explanation i understood what it was supposed to visualize. more cross highlighting between the different modules could make itmore clear.
- the design is bulky, the visualizations ar too packed and should be spreaded out. The nodes could have a proper name.
- the color choice is hard for color blind people
- no comments
- line chart could show different color to idéntify thé nodes and be in the same order.

Appendix 8. Consent form used for the front-end user based evaluation

Informed consent form

Title research:
Responsible researcher:

To be completed by the participant

I declare in a manner obvious to me, to be informed about the nature, method, target and [if present] the risks and load of the investigation.

I know that the data and results of the study will only be published anonymously and confidentially to third parties. My questions have been answered satisfactorily.

[If applicable] I understand that film, photo, and video content or operation thereof will be used only for analysis and / or scientific presentations.

I voluntarily agree to take part in this study. While I reserve the right to terminate my participation in this study without giving a reason at any time.

Name participant: ………

Date: ……………….………….… Signature participant: ……………………………………………………….……

To be completed by the executive researcher

I have given a verbal and written explanation of the study. I will answer remaining questions about the investigation into power. The participant will not suffer any adverse consequences in case of any early termination of participation in this study.

Name researcher: ………

Date: ………………………………… Signature researcher: ……………….……………………………………….

Appendix 9. Consent forms signed by the participants

Informed consent form

Title research: Ensketemp - usability
Responsible researcher: Yoann M.E. Latzel

To be completed by the participant

I declare in a manner obvious to me, to be informed about the nature, method, target and [if present] the risks and load of the investigation.

I know that the data and results of the study will only be published anonymously and confidentially to third parties. My questions have been answered satisfactorily.

[If applicable] I understand that film, photo, and video content or operation thereof will be used only for analysis and / or scientific presentations.

I voluntarily agree to take part in this study. While I reserve the right to terminate my participation in this study without giving a reason at any time.

Name participant: Luuk Renders

Date: 6/3/18 Signature participant:

To be completed by the executive researcher

I have given an spoken and written explanation of the study. I will answer remaining questions about the investigation into power. The participant will not suffer any adverse consequences in case of any early termination of participation in this study.

Name researcher: Yoann M.E. Latzel

Date: 06/03/2018 Signature researcher:

Informed consent form

Title research: Ensketemp - usability
Responsible researcher: Yoann M.E Latzel.

To be completed by the participant

I declare in a manner obvious to me, to be informed about the nature, method, target and [if present] the risks and load of the investigation.

I know that the data and results of the study will only be published anonymously and confidentially to third parties. My questions have been answered satisfactorily.

[If applicable] I understand that film, photo, and video content or operation thereof will be used only for analysis and / or scientific presentations.

I voluntarily agree to take part in this study. While I reserve the right to terminate my participation in this study without giving a reason at any time.

Name participant: Julia Pühl

Date: 6/03/18 Signature participant: Pühl

To be completed by the executive researcher

I have given an spoken and written explanation of the study. I will answer remaining questions about the investigation into power. The participant will not suffer any adverse consequences in case of any early termination of participation in this study.

Name researcher: Yoann M.E. Latzel

Date: 06/03/18 Signature researcher:

Informed consent form

Title research: Ensketemp - usability
Responsible researcher: Yoann M.E. Latzel

To be completed by the participant

I declare in a manner obvious to me, to be informed about the nature, method, target and [if present] the risks and load of the investigation.

I know that the data and results of the study will only be published anonymously and confidentially to third parties. My questions have been answered satisfactorily.

[If applicable] I understand that film, photo, and video content or operation thereof will be used only for analysis and / or scientific presentations.

I voluntarily agree to take part in this study. While I reserve the right to terminate my participation in this study without giving a reason at any time.

Name participant: Marco Gr[illegible]

Date: 06/03/2018 Signature participant:

To be completed by the executive researcher

I have given an spoken and written explanation of the study. I will answer remaining questions about the investigation into power. The participant will not suffer any adverse consequences in case of any early termination of participation in this study.

Name researcher: Yoann M.E. Latzel

Date: 06/03/2018 Signature researcher:

Informed consent form

Title research: Emsketemp - usability
Responsible researcher: Yoann M.E. Latzer

To be completed by the participant

I declare in a manner obvious to me, to be informed about the nature, method, target and [if present] the risks and load of the investigation.

I know that the data and results of the study will only be published anonymously and confidentially to third parties. My questions have been answered satisfactorily.

[If applicable] I understand that film, photo, and video content or operation thereof will be used only for analysis and / or scientific presentations.

I voluntarily agree to take part in this study. While I reserve the right to terminate my participation in this study without giving a reason at any time.

Name participant: U d Wande.

Date: 6 mrt 2018. Signature participant:

To be completed by the executive researcher

I have given an spoken and written explanation of the study. I will answer remaining questions about the investigation into power. The participant will not suffer any adverse consequences in case of any early termination of participation in this study.

Name researcher: Yoann M.E. Latzer

Date: 06/03/2018 Signature researcher:

Informed consent form

Title research: Emsketemp - Usability
Responsible researcher: Yoann M.E. Latzer

To be completed by the participant

I declare in a manner obvious to me, to be informed about the nature, method, target and [if present] the risks and load of the investigation.

I know that the data and results of the study will only be published anonymously and confidentially to third parties. My questions have been answered satisfactorily.

[If applicable] I understand that film, photo, and video content or operation thereof will be used only for analysis and / or scientific presentations.

I voluntarily agree to take part in this study. While I reserve the right to terminate my participation in this study without giving a reason at any time.

Name participant: Harry Collins

Date: 6th March 2018 Signature participant:

To be completed by the executive researcher

I have given an spoken and written explanation of the study. I will answer remaining questions about the investigation into power. The participant will not suffer any adverse consequences in case of any early termination of participation in this study.

Name researcher: Yoann M.E. Latzer

Date: 06/03/2018 Signature researcher:

Informed consent form

Title research: Emsketemp - usability
Responsible researcher: Yoann M.E. Latzer

To be completed by the participant

I declare in a manner obvious to me, to be informed about the nature, method, target and [if present] the risks and load of the investigation.

I know that the data and results of the study will only be published anonymously and confidentially to third parties. My questions have been answered satisfactorily.

[If applicable] I understand that film, photo, and video content or operation thereof will be used only for analysis and / or scientific presentations.

I voluntarily agree to take part in this study. While I reserve the right to terminate my participation in this study without giving a reason at any time.

Name participant: Radhika Kapoor

Date: 06.03.2018 Signature participant:

To be completed by the executive researcher

I have given an spoken and written explanation of the study. I will answer remaining questions about the investigation into power. The participant will not suffer any adverse consequences in case of any early termination of participation in this study.

Name researcher: Yoann M.E. Latzer

Date: 06/03/2018 Signature researcher:

Appendix 10. Temperature poster mockup idea