Web-based Visualization for Exploring Public School Data

Aron Baldvinson 201708851
Alex Krogh Smythe 201708400
Anders Wiggers 201708085

Bachelor Report (15 ECTS) in IT Product Development
Advisor: Hans-Jörg Schulz
Department of Computer Science, Aarhus University
September 2020

AARHUS
UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Abstract

The Danish government encourages all public schools in Denmark to provide as many statistics as possible to The Danish Ministry of Education and Research to store, analyze, and publish on uddannelsesstatistik.dk. Navigating the aforementioned website and exploring the data provided is a labor-intensive and a non-user-friendly task. The data is spread across multiple sites and difficult to work with. By identifying the users' requirements in the context of searching for information on public school data, we discovered a demand for a platform that allows users to easily navigate and explore such data. In the process of conceptualizing such an application, we found the literature insufficient in terms of presenting a general solution to visualize heterogeneous data sets to a broad user group. From this we formulated our hypothesis: Can we solve to the challenge of visualizing heterogeneous data sets to a broad user group by developing and implementing a framework that generalizes a solution the challenge. This resulted in the creation of the DEEVA Framework, that aims to support developers in the process of building visualization applications. To confirm or disprove the hypothesis, we created an application with a user-friendly interface that facilitates data exploration with a focus on the iterative process. Results from the evaluation confirmed our hypothesis. However further iterations of the framework are needed to fully realize its potential.

A link to a video-walkthrough of the application - https://bit.ly/2XHM2Ic.

Acknowledgements

We would like to thank our advisor, Hans-Jörg Schulz, for being very helpful and an inspiration throughout the thesis. The project would not have been the same without you - Thank you!

Contents

Chapter 1

Introduction

All public schools in Denmark are encouraged to collect and report a large number of statistics that ranges all the way from average grades of entire classes to socioeconomic status of students from individual countries. This data may be extremely valuable to user groups such as researchers, teachers, students, parents, politicians etc. The data has the potential to allow politicians to explore how well a certain funding has led a school to progress over time. It may also allow for parents to determine if they want to purchase real estate in one specific part of town based on the schools in the area. However, users are only able to perform such actions if the data is presented in a way that reinforces their intentions.
Our goal is to create an application that visualizes this data and two main challenges arise: The first challenge is that the data has to be gathered, processed, and prepared for delivery. It is hard to create a general solution to this as no data gathering process is the same, however in this case it involves DOM-manipulation, downloading files, and scraping using python. The entire process can be read in Section 4.3. The other challenge is to create a user interface that supports the end goals of many different users. The solution to this may be read in Section 4.6.

From this one main question arises: *In which way can we visualize a heterogeneous data set, while simultaneously satisfying as many different user groups as possible?* This research question originated from an investigation into similar projects such as HomeFinder (Weng et al., 2018) and NameVoyager (Wattenberg and Kriss, 2006), that have faced some of the same challenges as us. Instead of creating a unique solution to our project, we hypothesize that we can solve the challenge by developing a framework that presents a general solution.
The solution is The DEEVA Framework consisting of 5 different phases, each implementing different concepts from the data visualization field as described in Section 3.4. Based on a user requirement analysis in Section 3.2, the framework helps developers to envision and create visualization applications that face the same challenges as this project. To test the usability of the framework, we developed a visualization application using the DEEVA Framework and performed a series of evaluations using real users. Through the user tests in Section 5.1.2, we found our hypothesis to be confirmed and that there is in fact a way to solve the complex challenge of developing a visualization

application that needs to satisfy many different user groups with a general solution. However we found that to harness the full potential of the DEEVA Framework, further iterations of are needed.

Chapter 2

Related Work

The following chapter will introduce and review related work that explores the current field of data visualization. We examine what techniques and practices others have used in similar projects and introduce terms used throughout the literature. The related work section is structured into three main categories: Data Preprocessing, Data Visualization and Interaction. Each category has relevant subcategory that highlights the techniques and concepts.

2.1 Data Preprocessing

2.1.1 Data Cleaning

When viewing data it is often preferable to view the raw data as it provides the full in-depth view of the set. However when viewing the data as is, problems such as missing data points in the sets, becomes very apparent. Data cleaning is the act of making the data ready, filling in missing values and normalize the set(Ward et al., 2010).

Transformation

Transformation of data is the process of transforming data with inconsistencies into a uniform format (Raman and Hellerstein, 1999). These inconsistencies can consist of bad data formatting, schema errors, misspelling, missing information or invalid data.

Multiple Imputation

A useful strategy for handling data sets with missing values or sets that are incomplete is using imputation. Imputation is used to fill in the missing values with plausible values (Schafer, 1999). Rubin (2004) introduced an accurate way to impute missing data with a technique called Multiple Imputation where repeated imputation is used to get a more accurate representation.

Normalization

Normalization (Ward et al., 2010) is a technique to convert the data in such a way that it allows for comparison to be drawn between seemingly unrelated values. The

feature enables the comparison between data sets that otherwise would have been incompatible. An attribute of this is that explorability for the user is increased as the possible combination of data set comparisons is drastically increased.

2.1.2 Data Reduction

Data Reduction is the process of identifying and diminishing unnecessary details in data sets in order to highlight the characteristics and essentials of the data. This is important when working with large data sets that keep increasing in size to avoid computational inefficiencies and cluttering of the visualizations (Cui et al., 2006). Cui et al. (2006) identifies techniques for data Reductions such as clustering and sampling.

Sampling

Dix and Ellis (2002) suggest that random sampling can make the visualization of large data sets more efficient and perceptually effective. They describe three scenarios where using randomness in sampling can enhance visualization.

- **Aggregates:** If the visualization is a sum or aggregate of a statistic, then a histogram, pie chart etc. of random data points within the data set will reflect the full data set if enough samples are drawn.
- **Point And Line Data:** A plot with enough data points may begin to simply turn the graph into a solid. Random sampling works as a counter measure to this since the characteristics and patterns will appear before the graph turns incomprehensible.
- **Individual Data Items:** The most important or significant data entries can be shown in full detail while the lesser relevant data entries are randomly sampled at shown upon zooming in.

Clustering

Clustering is the technique of grouping data into related sets of observations or clusters with similar attributes (Myatt, 2006). Clustering is often used when little information is available about the data. Hundreds of powerful clustering algorithms have been created to help categorize and automate the process of clustering large sets of data (Keke Chen and Ling Liu, 2003). Keke Chen and Ling Liu (2003) acknowledge the fact that human participation is inevitable when utilizing any clustering algorithm, since the clusters require analysis and understanding after the algorithm has done its work. Keke Chen and Ling Liu (2003) have therefore developed a visual framework (VISTA) that encourages human participation in the clustering process, allowing users to steer/monitor and refine the clustering process with domain knowledge.

Aggregation

Aggregation (Ward et al., 2010) is used as a data reduction method to gather data points from a vast data set making it suitable for visualization. The technique groups data based on their similarity and then represents the group by some amount or mean of the

data. There are two points to aggregation: the grouping of data and the representation of the data. Liu et al. (2013) groups data in predefined bins. A bin is a data structure that allows efficient region queries. Afterward the density of each bin is defined as number of data points falling within those bins. When the data is aggregated this way, global patterns can be made easily observable, while simultaneously enabling multiple levels of resolution of the bin sizes. This feature allows users to explore the data set and drill down on specifics whilst also being able to see a full view.

2.2 Data Visualization

2.2.1 Animation

Animation in data visualization can be used in situations where the users view is changing. Instantaneous cuts between views makes the context difficult for the user to understand. There is also an aesthetic side to animation that makes data visualization more enjoyable. (Gonzalez, 1996)

Animation in Scatterplots

Elmqvist et al. (2008) address some of the shortcomings of visualizing multidimensional data in scatter-plots namely the limit of how many data dimensions can be represented at a time. The standard solution is to only visualize a subset of the data set dimensions and relying on the user to control the visual mapping. This creates a lack of structure in the visual exploration process and fails to show the relations between the subsets of the data. Elmqvist et al. (2008) proposed a design as seen in Figure 2.1a that addresses the shortcoming mentioned above and integrate several components that support the basic tasks of comparing and correlating dimensions in multidimensional scatterplot data sets. These components include a design for structured exploration in data using 3D animated transitions.

Animation When Zooming

Users often experience instantaneous switching between views which can seem very distracting and cause loss of the mental image. Cockburn et al. (2009) suggests using animation to transition through the pre-zoom and post-zoom state. They suggest having 0.3 to 1 seconds of animation/transitioning between the two states. An example of this is google earth when the user is searching for a city across the globe as can bee seen in Figure 2.1b

(a) (Elmqvist et al., 2008) (b) (Cockburn et al., 2009)

Figure 2.1: On the left an animation between one scatterplot dimensions to another and animation when zooming on the right.

2.2.2 Multiple Coordinated Views

Multiple coordinated views (Roberts, 2007) are a specific exploratory visualization technique that enables users to explore their data. By displaying different views that react to each others input, users are able to explore data in different ways. Users are able to compare data from the different views and adjust filters or selections based on what the different views are displaying.

Filtering Search Results

Homefinder Revisited (Weng et al., 2018) uses multiple coordinated views to help users narrow down possible housing opportunities by guiding the users through the stages of choosing a home with different views. View A-E in Figure 2.2a lets the users explore and apply constraints to the view. View F-J are on the mini map and allows users to explore aspects such as location and distance. The views from K-M allows users to filter and compare different housing opportunities and view N lets users save a specific location. In that way the MCV acts as a pipeline to support the users intentions.

Pattern Exploration

PRISMA (da Silva Kauer et al., 2008) provides users the ability to interact and find patterns within a data set using multiple coordinated views. The multiple coordinated views allow the users to highlight and filter through the data set, where the results of such filtering are reflected in the other views which can be been seen in Figure 2.2b

(a) (Weng et al., 2018)

(b) (da Silva Kauer et al., 2008)

Figure 2.2: Display of Multiple Coordinated Views: HomeFinder Revisted on the left and PRISMA on the right.

2.2.3 Overview + Detail

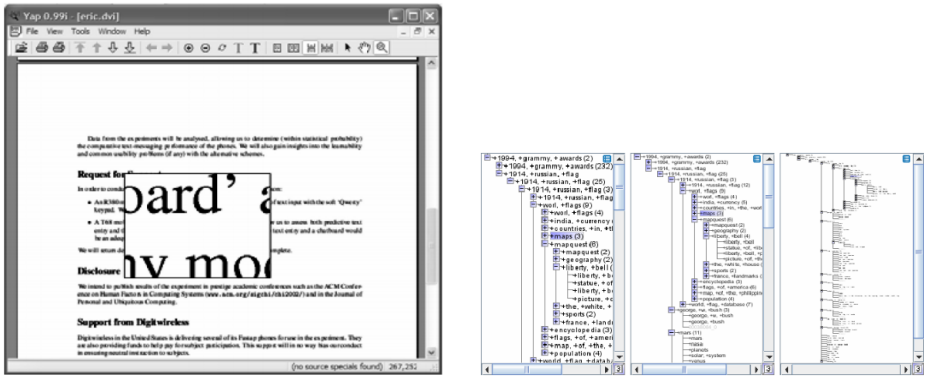
An Overview + Detail interface is normally identified by displaying an overview and a detailed view either simultaneously or subsequently (Card et al., 1999). For data exploration Overview + Detail can be particularly useful as it allows users to mentally map the data and reduces the short term memory load.

Interactive Lenses

Interactive Lenses are an example of an Overview + Detail technique that separates the views on the z-axis by adding a frame at the cursors location which lies on a separate plane from the content (Cockburn et al., 2009). Lenses have typically been used as *"Magnification metaphor in standard desktop environments since the early 1990s."*(Cockburn et al., 2009, p. 7), however, lenses can take many forms and have many functionalities. An example of this type of Overview + Detail can be seen in Figure 2.3a, which is a previewer that magnifies a detailed region shown in a lens.

Fisheye Tree Views Pt. 1

Fisheye tree views (Tominski et al., 2006) is a way to create Overview + Detail in a standard tree view. The issue with standard tree views, is that when users are opening menus they tend to force either the horizontal or the vertical axis out of view.
Tominski et al. (2006) faces this issue by applying a standard mouse-scroll zoom that allows users to see the entire tree view and zoom in to achieve a more detailed view of the menu as seen in Figure 2.3b. By doing so users are now able to maintain the mental map with the overview and get the more detailed view of the tree, thereby freeing mental resources which can be allocated to fulfilling their primary task.



(a) (Cockburn et al., 2009) (b) (Tominski et al., 2006)

Figure 2.3: Display of Overview + Detail interfaces.

2.2.4 Focus + Context

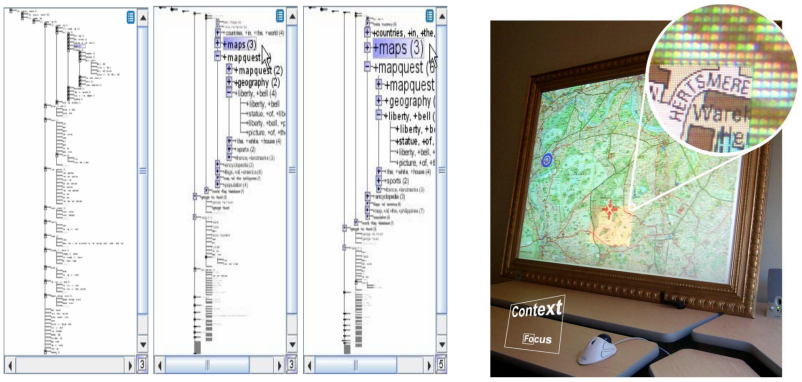
A Focus + Context interface (Card et al., 1999) is a way to seamlessly integrate both the focus and the surrounding context into a single view. This attribute makes it easy for a user to explore data relationships as the focus and underlying context is merged. The aim is to reduce short term memory load and increase the users *"ability to comprehend and manipulate the information."* (Cockburn et al., 2009, p. 11)

Fisheye Tree Views Pt. 2

Fisheye tree view (Tominski et al., 2006) is a Focus + Context solution to the problems that may arise from Overview + Detail. The technique is achieved by enlarging the selected menu item and its surrounding neighbours to a lesser degree. The selected item, which is the focal point, is seamlessly integrated with the surrounding context that allows the user to understand the context of the highlighted menu item. This attribute promotes exploration of the nearby data as information that lies far away is diminished as seen in Figure 2.4a.

Wide Field of View Systems

Wide Field-of-View system (Cockburn et al., 2009) is a type of focus + context technique where a small high resolution focal area view is combined with a large low resolution view to display context. Field-of-View can be used in head up displays, wall projections or on normal screens. Baudisch et al. (2002) has created an example that utilizes the techniques to stitch a high resolution image together with a low resolution image of a map as seen in Figure 2.4b



(a) (Tominski et al., 2006) (b) (Baudisch et al., 2002)

Figure 2.4: The left image illustrates a tree view using Focus + Context functionality and on the right is a Field of View system with a mixed resolution large display.

2.3 Interaction

2.3.1 Dynamic Queries

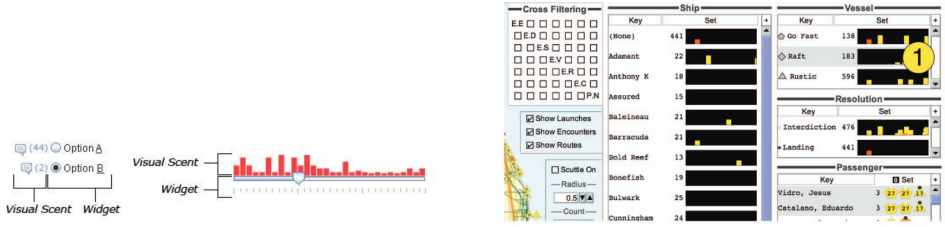
Dynamic Queries (Card et al., 1999)(Ahlberg et al., 1992) is an alternative technique to SQL for querying data from databases. It is more effective for the novice user as it takes very little time to learn compared to SQL queries and is *"a natural method for requesting data when the output is going to have a visual form"* (Card et al., 1999, p. 235). Dynamic queries are a form of direct manipulation for database queries. Direct manipulation of data has proven to be successful in other application such as display editors and graphical environments. The ability to directly manipulate the queries allows the users to query data at a greater speed increasing explorability.

Scented Widgets

Scented Widgets (Willett et al., 2007) are a form of dynamic query which increases usability in information spaces. The widgets use embedded visualizations on top of graphical user interface controls to enhance exploration of data sets. A scent is generated from social data of other users behavior or data-driven metrics. This scent lowers the "cost" of information gathering and improves exploration of the set. New users can then use the scent to navigate the set. Figure 2.5a displays the visual scent built on top of a widget.

Cross-filtering

Cross-filtering (Weaver, 2010) is a method that allows for direct manipulation of the queried data. Furthermore the filtering allows for fast and flexible research into relationships that may be spread across a number of data sets. This allows the user to explore relationships in multidimensional data sets, that would otherwise have been hard to discover as seen in Figure 2.5b.



(a) (Willett et al., 2007) (b) (Weaver, 2010)

Figure 2.5: Left figure illustrates a scented slider widget with a visual scent in the form of a histogram. The figure on the right illustrates a cross filtering interface.

2.3.2 Interactive Lenses

Interactive lenses (Tominski et al., 2017) assists the user with visualizations by temporarily altering the current view of an existing data visualization within the frame of the lens. Interactive lenses can change or manipulate a display in many ways such as magnification of an area or giving context to the user. A lens with a magnifying property needs to react slowly to not confuse the user with information overload, where other lens-properties might need to react with the fast re-positioning that the mouse enables.

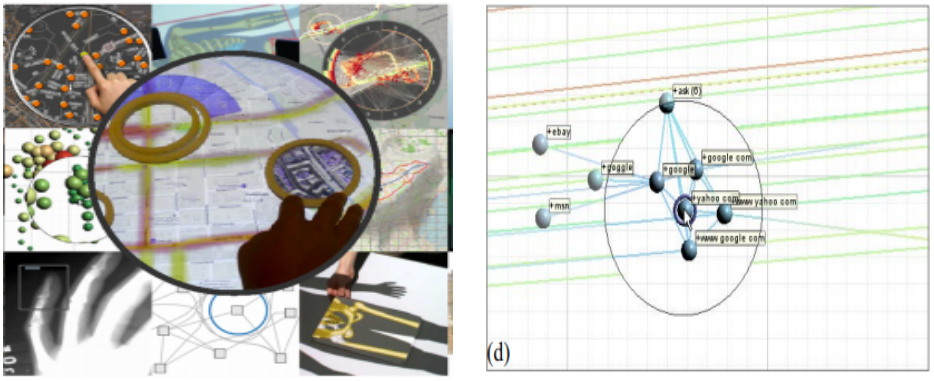
Different Modalities

Interactive lenses take advantage of an array of different interaction modalities and shows how a various palette of lenses can be used. A few can be seen in Figure 2.6a. These modalities include mouse and keyboard interaction, touch and multi-touch interaction, tangible interaction, tangible view and spatial interaction and lastly gaze-based interaction & head tracking.

Lenses for Graph Visualization

Graphs can easily become cluttered depending on how many data entries it consists of. Therefore users may lose their ability to explore the data because of the cognitive load. Tominski et al. (2006) introduces three lenses to face this issue as seen in Figure 2.6b.

- **Local Edge Lens:** A lens that only highlights the edges between the nodes within the lens. This helps and prevents cluttering.
- **Bring Neighbors Lens:** A lens that brings the surrounding nodes close to the lens in order to help the user maintain the mental image of the graph.
- **Composite Lens:** The combination of the Local Edge Lens and the Bring Neighbors Lens.



(a) (Tominski et al., 2006) (b) (Tominski et al., 2017)

Figure 2.6: Illustration of a Composite Lens on the left and collage of different lenses on the right.

2.3.3 Brushing

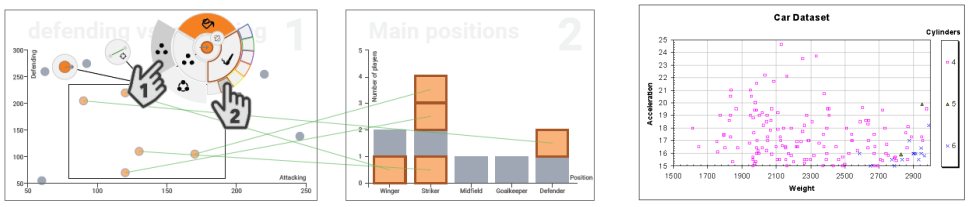
Brushing has become prevalent in almost every interactive visualization environment especially for data exploration and analysis in multiple coordinated views (Roberts and Wright, 2006)(Chaoran and Hauser, 2019). Brushing is in its simplest form a technique where users have the ability to explore the relationships between selected data points in a plotted graphic and instantaneously view other representations of that data. This gives an insight into how the visualization is linked with other displayed data, which offers the user a valuable investigative tool for data exploration.

Brushing and Linking with Personal Agency

Extensions on the brushing techniques can be seen for instance in MyBrush (Koytek et al., 2017), that takes the brushing technique and couples it with personal agency (Coyle et al., 2012) and linking. Koytek et al. (2017) created an interface that lets users query data with direct manipulation emphasizing on providing direct feedback as seen in Figure 2.7a. The main objective of MyBrush is to support visibility of brush configurations, provide activity awareness and encourage people to explore the possibilities of data analysis. As a result the MyBrush tool offers data exploration, learnability and increases the sense of control for its users.

Ubiquitous Brushing

Typically when using the brushing technique users are limited to brush over a plotted area. Roberts and Wright (2006) explores the concept of Ubiquitous brushing, allowing users to not only brush the plot area, but also context-giving components (legend, axis, labels, menus etc.). Resulting in an interactive technique that allows users to brush over any part of the interface and increase the efficiency of selecting elements as seen in Figure 2.7b (Roberts and Wright, 2006).



(a) (Koytek et al., 2017) (b) (Roberts and Wright, 2006)

Figure 2.7: An illustration of Mybrush on the left and Ubiquitous Brushing on the right.

2.3.4 Zoomable UI

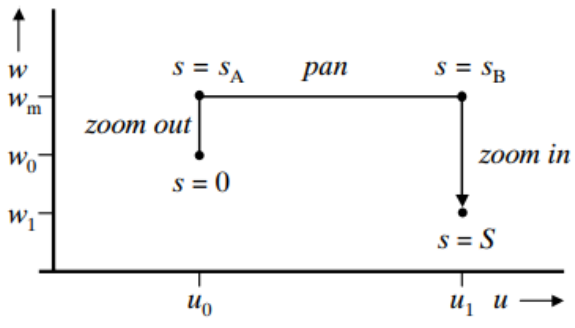
A Zoomable UI (ZUI) is a technique that allows users to interactively magnify(zoom in to focus) or demagnify(zoom out for context) as a method of navigation on data sets (Cockburn et al., 2009). Allowing users to change the scale at which the information space is viewed with zooming and allowing users to change the area of the information space visible through panning (Hornbaek et al., 2002).

Smooth Zooming And Panning

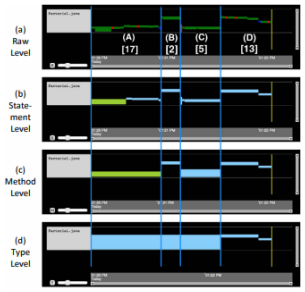
Encountering large information spaces like 2d maps, images or even abstract visualization requires viewing data at different levels of detail (van Wijk and Nuij, 2003). Users interacting with these large data sets benefit from having the ability to inspect detail and maintain an overview of that data, allowing them to understand context (Shneiderman, 1996). van Wijk and Nuij (2003) explore defining metrics that helps developers achieve a smooth and efficient zooming and panning from one view to another as seen in Figure 2.8a.

Semantic Zooming

An extension on ZUI is the incorporation of semantic zooming. Semantic zooming is a technique that takes a representation of a object and adapts to the number of pixels available in an image-space region occupied by the object. Allowing images to change subtly or dramatically based on the scale (Munzner, 2015). Yoon and Myers (2015) utilize semantic zoom to offer users higher-level of abstraction on a zoomable timeline displaying real-time code changes as seen in Figure 2.8b . Dividing the timeline into four levels of abstraction giving users control of their preferred view via zooming. Resulting in aiding users to understand and interact with the timeline and avoid information overload.



(a) (van Wijk and Nuij, 2003)



(b) (Yoon and Myers, 2015)

Figure 2.8: An illustration of the four levels of abstraction on the left and a diagram on the right illustrating how to achieve a smooth transition between views.

Chapter 3

Concept

3.1 Introduction

This chapter firstly introduces the different steps taken to complete a requirement analysis for a data visualization application based on public school data in Denmark. The methods used are mostly human-centered design techniques along with a questionnaire.

Human-centered design embraces the users and allows us to achieve a greater understanding of the users real needs instead of their perceived needs. Conducting a questionnaire is a good way to collect quantitative data that can provide information about a bigger group of people than we can by conducting individual interviews. It turns out that the number of different user groups is much bigger than originally anticipated based on the results from the inquiries made in the requirement analysis below.

This introduces the challenge of creating an application that can accommodate a large variety of users and support many different goals. This chapter will describe the DEEVA Framework that was developed to solve this issue. The framework allows developers to gain insight into how they should structure a data visualization application and which concepts from the visualization field would be ideal to use during the 5 phases, that are the building blocks of the framework that will be further described in Section 3.4.2. The DEEVA Framework Section after the requirement analysis dives deeper into the specifics of these different phases and argues for the theory behind it and why it works the way it does.

3.2 Requirement Analysis

To gather the needs and requirements for as many users as possible the users need to be identified. This is to avoid developing a platform that does not accommodate the needs of the actual users. To do this a PACT-analysis (Benyon, 2014) that highlights People, Activities, Contexts, and Technologies was conducted. We created four personas based on our findings from the analysis as well as a questionnaire to further explore and identify requirements. The personas will be used as a guideline and as a benchmark for our further development. We will specifically use the personas and their diversity to

generate a journey map that may reveal opportunities and bottlenecks that we have to be aware of when designing the user interface and the user experience.

3.2.1 PACT-Analysis

The objective of the PACT-analysis is to highlight stakeholders and to understand the users that are engaged with a platform like this. PACT helps to identify differences in People (social and psychological), Activities (usability, temporality, and complexity), Context (physical and social settings) and Technologies (I/O and architecture).

People

The users consist of parents, journalists, politicians and whoever wants to explore the data for other reasons, which in other words could be anyone. Parents might use the application to explore their children's schooling opportunities, whereas politicians and journalists might use the platform to identify patterns or issues in the schooling system based on the result of different schools. People of all shapes and sizes are able to physically use the platform since it is a web-application. However elderly people might have an issue interacting with the platform because of their inexperience with technology. At last it is expected that people with various disabilities might use the application as well, so the platform needs to be up to date regarding usability and accessibility measures such as color schemes that support colorblindness, etc.

Activities

The main goal of engaging with the platform may vary from user to user. The one thing they all have in common is the need engage with a very large data set that is based on public school data. To allow the users to find the information they need means that the interactions between the user and the platform must support exploration. Users need to be able to fine-tune their searches efficiently and maintain their mental map of the data set as well. It is also important to acknowledge the *different* ways that users might engage with the application. Parents might want to highlight and filter by certain areas or qualities to get a list of top 5 performing public schools within a region, where journalists and politicians might want to focus on certain schools development over time.

Context

The physical context of the users is unpredictable since the platform will be available on portable devices. When users will be engaged with the application depends on parameters such as the users' profession. Some users may be working a 9-to-5 job and only have time to use the application during the afternoon, where some users may be using the application as a part of their job to do research. It is therefore important that users can save their searches and results to return to their research and exploration at a later time.
Social context may also vary within the users and their goals. If users are exploring the data to find a suitable place to settle down, then multiple people may be looking at the

screen together. There is also the case where users are exploring the data at the same time but using different computers or located at different physical locations. In that case users must be able to share their findings perhaps by sharing a link.

Technologies

Users must be able to filter, search, and interact with the a large number of data sets intuitively. In order to display the data and to enable users to manipulate the data, we have to query it from a database hosted on a web server. People unfamiliar with database queries and fetching data should still be able to get the full experience of the application. The more technical aspects of the system architecture will be further explored and elaborated in Chapter 4.

3.2.2 Personas

The personas seen in Figure 3.1 are concrete representations of our users, their backgrounds, interests and goals of using the platform (Benyon, 2014). By defining the characteristics of the users with inspiration from our previous analysis, we have collected enough information to define the diversity of our user group. These characteristics highlight the goals and aspirations of the users when using our platform.

Figure 3.1: An illustration of the four personas.

3.2.3 Questionnaire

User-centered design is impossible to do without actually including the users at one point (Vosough et al., 2017). A questionnaire was therefore conducted in order to gather and prioritize the user's requirements systematically. A questionnaire is an effective way to gather large amounts of data in a quantitative way (Benyon, 2014).
The questionnaire collects information about the users' age, occupancy, parental status, and to which degree the schools in the vicinity had an influence on their place of residence. Users are also able to mark which features and information that they might be interested in. The overall goal of the questionnaire is to pinpoint which features are useful to the users and to highlight any shortcomings in the PACT-analysis and the personas.

Results

These are the results from the questionnaire, for the full list of results see Appendix A.

- The average age of participants is: **33 years old.**
- The occupancy of participants ranges from: **Students to Retirees**.
- Parental status: **55.6% with no children** and **44.4% with children**
- Device preferred for working on the platform:
  PC or MAC: **69.4%.**
  Smartphone: **27.8%**
  Tablet: **2.8%**
- Influence of nearby schools when purchasing real estate as seen in Figure 3.2
- A list of most requested information by the participants as seen in Figure 3.3.
- A list of the most requested features by the participants as seen in Figure 3.4.

Scale	1	2	3	4	5
Influence of purchase	11.1%	16.7%	33.3%	27.8%	11.1%

Figure 3.2: Scale ranging from 1-5, 1 being no influence at all, 5 being highly influential

Information	Percentage
Students well-being	86.1
Competence coverage	50
Number of teacher for each student	41.7
Number of students applying for further studies	41.6
Mean grade for given school	38.9
Number of students per school	33.3

Figure 3.3: A summary of the most sought after information.

Information	Percentage
Compare schools with each other	69.4
Use the platform for research	50
Explore indivdiual	38.9
Create graphs with user determined parameters	27.8
Download the results	25

Figure 3.4: A summary of the most wanted features.

3.2.4 Journey Maps

To conceptualize the best journey for the user, a journey map of the pipeline was created as described by Benyon (2014). The procedure of creating the journey map is as follows: (1) Identify the main goal(s) of the application. (2) Gather all the different interactions we want the users to experience or to be a part of the journey throughout the application based on the questionnaire. Lastly sort the different interactions and moments into a sequence to create the best pipeline for the user. The full illustration of the User Journey Map can be seen in Figure 3.6.

Main Goal

The main goal of using our application depends on the context of the user. We have identified a number of objectives based on our personas, PACT-analysis, and the questionnaire. Some of the users' goals will be observing statistics in the form of graphs to do research, while other users will be looking for specific schools by comparing them individually. Some might be purchasing real estate in the vicinity, some might be looking for work and some might just be using the application for entertainment or out of curiosity. Instead of compromising the end product or to only acknowledge one specific group of users, our goal is to create a solution that accommodates multiple user groups and their goals throughout the application.

Core Moments

We identified the following high-level core moments that all users will experience during their use of the application illustrated in Figure 3.5. They have been placed randomly as the moments aren't sorted in any meaningful way yet.

Features

Other features that users must be able to use are also included in Figure 3.5 below. These features are meant to guide the users to a point where they will experience the core moments identified above.

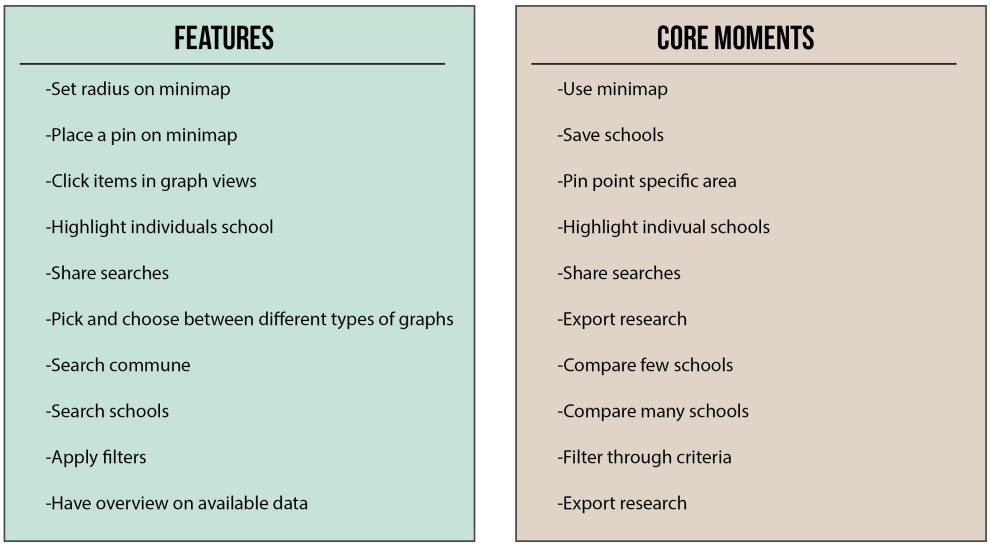


Figure 3.5: Core moments and other identified features

Defining The User Journey

So far we mapped out some core moments and features that people have shown interest in. In this section we combined all of these requirements into a pipeline in order to simulate the full user journey, as seen in Figure 3.6. The main goal of this task is to give us a full mental map of the application and how each requirement needs to be developed. The results from the journey map will lay the foundation for how we will design the application.

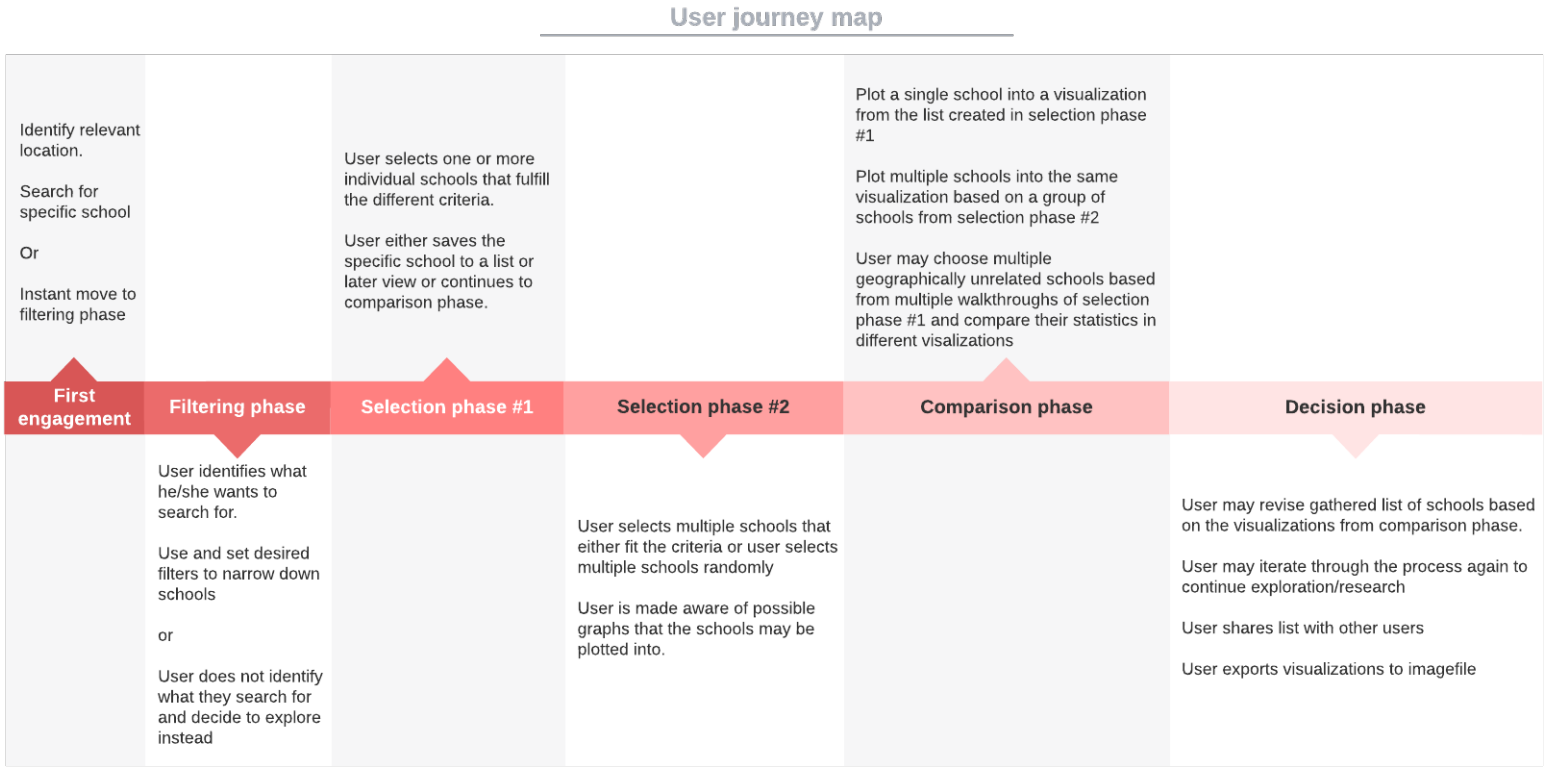


Figure 3.6: The journey map that illustrates the users path through the application.

3.3 Summary of Requirements

Throughout this chapter various user-centered design methods were used to accumulate requirements for the platform. The requirements gathered will be summarized in this section.

PACT

The PACT-analysis highlighted the need for a platform that can accomodate a variety of users with different goals. The PACT-analysis also helped to identify users with disabilities such as vision impairments.
Time to complete exploration and research may vary and the PACT-analysis has also highlighted that users may have the need to save their search results in order to return to their work later.
Making sure that novice users still have the ability to take full advantage of the application is vital because everyone using it will be a novice user at first. Lastly the analysis provided insight into the experience that all users have in common: they will all be exploring the same large data set, so users must be able to search and filter through the data set in an easy way.

Personas

The personas emphasize the requirements extracted from the PACT-analysis, specifically the importance of accommodating a large variety of users. Ensuring no matter the agenda, all users should be able to achieve their intended goals when using the application. The personas will also be used as a reference for future decisions.

Questionnaire

The questionnaire provided a list of information and features that the application needs to provide. Specifically the well being of students, the number of students per school, number of students per teacher were valuable. Some information that the users showed interest in ranked much higher than expected (with an overwhelming percentage of 86.1%), namely students well being.
Features that the participants were mainly interested in was the ability to compare different schools with each other, explore individual school data and create their own charts with different personalized parameters. Participants who were interested in using the application for research wanted the ability to download the results gathered.

Journey Map

The journey map has taken all the gathered requirements and merged them into a full user experience. It has provided us with a mental map that has been separated into different phases where each phase fulfills some of the requirements set by the users. The user journey map gives each requirement context and gives a perspective into how all of the requirements should be linked.

Conclusion of Requirements

Based on the requirement analysis we identified numerous user groups where each user group has its own list of requirements. Because of the many different users that an application like this has to accommodate it is very difficult to create a simple application-structure or pipeline of interactions that satisfies each and every user. During our research we noticed other related projects such as Weng et al. (2018) and Wattenberg and Kriss (2006), where users had the ability to explore data visualizations. Each of these projects needed to reinvent the wheel to create a platform that suited just a small variety of users.

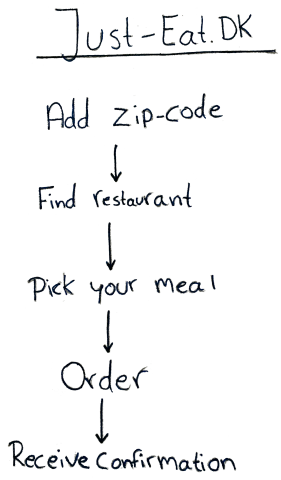
Because of the reasons above, we realized the need for a general framework to help develop and design exploratory focused data visualization applications. A framework that developers and researchers may utilize when facing a use case where user groups vary a lot and they all have different requirements and goals. The framework needs to provide a template for an interaction pipeline where developers must be able to interchange the different data visualization components depending on their own specific requirement analysis. The section below will introduce our process and the final framework.

3.4 The DEEVA Framework

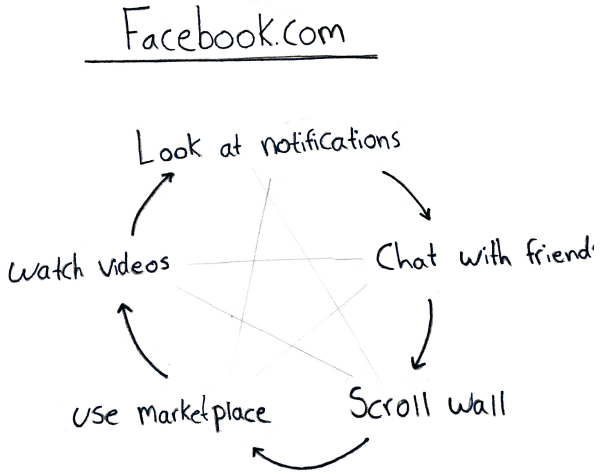
The final framework is called The DEEVA Framework and is an acronym of **D**eveloping **E**xploratory **E**nvironments in **V**isualization **A**pplications. The full framework is a combination of a structure that is based on Ben Shneiderman's mantra: "Overview first, then zoom and filter, details on demand" for data visualizations with a repetitive aspect that supports the exploratory element of the application as well as concepts from the data visualization field. These are merged into a high-level user journey that supports the goal of an exploratory process where developers can envision and implement their applications based on their own users and gathered requirements.

3.4.1 Theory behind DEEVA

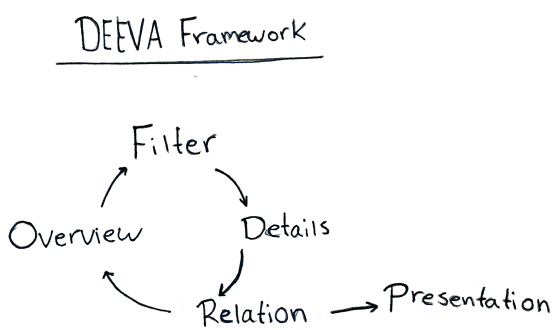
Our requirement analysis does not reflect a scenario where all users share the same goal. Figure 3.7 illustrates 3 possible high-level user journeys in different applications. Figure 3.7a is an example of an application that is based on a linear flow of action i.e when an application has a clearly defined start- and endpoint where all users share the same goal. An example of this is Just-eat.dk or any webshop out there. We say that a structure like this has a linear flow of action. On the other end of the spectrum we have the circular flow of action. The circular flow of action is often seen among social media e.g Facebook.com, where the user has no clearly defined beginning or end, as seen in Figure 3.7b.



(a) An illustration of an application with a linear flow of action, Just-eat.dk, that allows users to order food online from local restaurants based on their zip code. The application has a clearly defined start and end point with a user journey that points all users toward the exact same goal.



(b) An illustration of an application with a circular flow of action, Facebook.com, a social media site where users can connect with each other and utilize features such as videos, chatting, and buying/selling items. Users may start and end at any point and are able to do as many walkthroughs as they wish.



(c) A LO-FI illustration of the DEEVA Framework that highlights the combination of the linear and circular flow of action. By having the circular aspect to the framework, users can iterate multiple times throughout the application, which enables some users to accomplish their goal, while still allowing other users to continue into the Presentation Phase to complete a different set of goals. In that way users may exit the application in any given state when they have achieved what they wanted.

Figure 3.7: Three illustrations that illustrate the different types of action flow as well as the combination of the two, that we see in the DEEVA Framework.

The reason that we have chosen to use these specific sites and not any related visualization projects to illustrate the linear and circular flow, is to highlight that we are only speaking of navigation and user journey in the applications and not the functionalities. It also shows that this is a concept that is relevant in other application other than those in the visualization field.

The linear flow of action delivers a focus and an end goal to the application and an intuitive way to reach that goal. However the circular flow of action creates a way for users to perform tasks repeatedly and create a full experience of the different parts of the application. The DEEVA Framework's goal is therefore to create a logical journey throughout the application by combining the linear flow of action with a circular flow of action. This should create an exploratory environment while still allowing the user to reach an endpoint in the application which should accommodate many different users with different goals.

3.4.2 Structure and Use of DEEVA

As seen in illustration 3.7, the DEEVA Framework has merged the linear and the circular model into a single model. The model illustrated in the figure is called the combined model for obvious reasons. The combined model consists of five phases that the users must be able to go through during the use of a visualization application. The five different phases are all there to support the opportunities and goals that the different users might have. Each of the phases is a step with a high-level goal and developers should implement these goals using different concepts and techniques from data visualization field that suits the individual project best e.g concepts from the related work Chapter 2. An illustration of the DEEVA Framework with possible concepts and techniques is shown in Figure 3.8. Next we will introduce the different phases and argue why the final model works and looks the way it does.

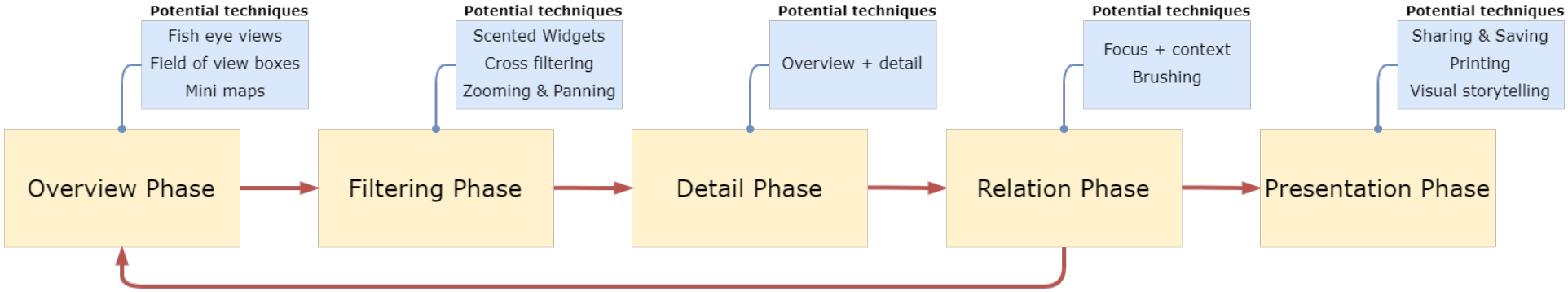


Figure 3.8: The Illustration shows how each phase relates to one another and how the output of one phase builds upon the next phase. The illustration also highlights the linear and circular aspects of the framework.

Overview Phase

The first phase in the framework is the Overview Phase. In the Overview Phase users must able to formulate their first intentions of using the given application. At this stage users should be able to create a mental map over the application and have the ability to deduct what the first interactions should be. An example of this could be to show the user which different fields it is possible to filter by.

The main idea of having the overview as the first phase is backed up by the famous mantra by Ben Shneiderman: "Overview first, zoom and filter, then details-on-demand" (Shneiderman, 1996), who argues that a taxonomy like this is useful to facilitate discussions and exploration. Examples of visualization techniques and that are compatible with this phase are fisheyes, field-of-view boxes, and zoomed out views such as mini maps. Most of the phases in this framework have been derived from the taxonomy that Ben Shneiderman suggests in the article "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations". A full comparison to the mantra is available in Section 3.4.3.

Filtering Phase

In this phase users are able to apply their first constraints to the data set. Two main possibilities arise when applying these constraints. Filtering allows users to either filter out facets of the data set that might be irrelevant to their searching or filter to include specific characteristics. A real-world example of this could be an automobile application where users are able to filter search results by applying a "No station cars, but everything else"-filter or a "station cars and nothing else"-filter.

Detail Phase

Users should at this point be able to choose a few items as a result of the filtering from the phase before this one. This is where a user may explore the low-level details about the results. An example of this could be clicking on a selected recipe in a cooking application where each ingredient is now revealed in detail. Analyzing this step, we see that users are now diverging their information space by being able to view details and discover the characteristics of the chosen item.

Relation Phase

The Relation Phase is the fourth phase of the framework. It allows users to put individual details into the context of other items and discover relationships and patterns between them. The Relation Phase is where the comparative aspect of a visualization application manifests itself. If the same example of the cooking application from the paragraph above is used, the Relation Phase may be the act of linking different ingredients to other recipes or discovering that certain ingredients are more commonly used in some parts of the world than others.

Presentation Phase

The Presentation Phase is the last phase that users must be able to go through while using a visualization application that follows this framework. The Presentation Phase allows users to extrapolate their findings in a presentable way. Users must be able to share, save, and export graphs, charts, and other views so the findings may be used later on. In Shneiderman's Data Type Task Taxonomy (TTT), he suggests that users should be able to share their filters as well. This may be done by creating a shareable link that automatically adjusts the controlling widgets to match the search results that the user wanted to share. It is also important that users can convey their findings from the Relation Phase in the presentations e.g by Brushing and Linking results in the shared views.

3.4.3 Comparing the DEEVA Framework to Shneidermans mantra

It is relevant to address the similarities and overlaps that the DEEVA Framework and Shneiderman's "Overview first, Zoom and Filter, Details on-demand" mantra has, as well as differences that we are trying to fix with this framework.
In the article, The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations, (Shneiderman, 1996), Shneiderman argues that there are 7 different data types to visualize (1-,2-,3-dimensional, temporal, network, tree and multi-dimensional) and the best way to visualize them is by utilizing his mantra. Shneiderman's intention behind the mantra is to highlight a way in which users get the best experience when working with *specific* data visualizations. An example of this could be the Tominski et al. (2006) article, where they utilize this exact mantra to discover the best way to visualize a tree view as seen in Figure 3.9. In other words, the mantra provides a solution on how to develop the user experience of one specific visualization at a time.

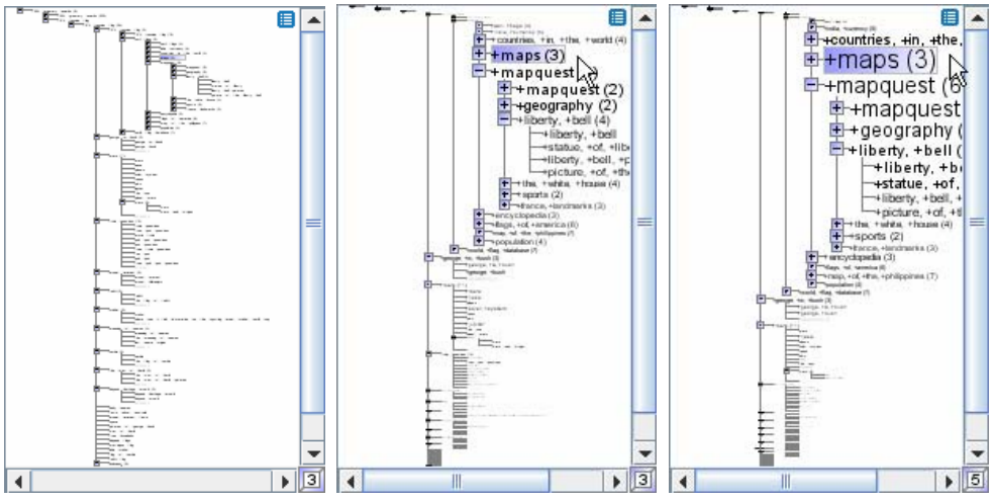


Figure 3.9: Illustration of how the mantra "Overview first, Zoom and Filter, Details on demand" manifests itself in a tree data type, Tominski et al. (2006).

The DEEVA Framework however makes use of the Shneiderman mantra by applying it to the user journey and thereby utilizes "Overview first, Zoom and Filter, Details on

demand" as a taxonomy throughout the application, instead of individual visualizations. Since our initial requirement analysis highlighted the many different goals that users have, the framework has been extended with the Relation Phase and Presentation Phase. This is because the Shneiderman mantra satisfies users that all share the same goal: visualizing the specific data type. However the user's goals of the visualization application ranges from just looking at simple statistics to detailed views and being able to find patterns within the school system all the way to publishing their results at the end.

The repetitiveness in the DEEVA Framework, manifested by the circular flow of action seen in Figure 3.7b, is not a part of the mantra either, but is an important element in the framework, as it catalyzes the iterative process.

Even though the main difference between the Shneiderman mantra and the DEEVA Framework is the taxonomic use of "Overview first, Zoom and Filter, Details on demand", it is also important to note that the applications that implements the DEEVA Framework are still able to use the mantra in their visualizations. However this depends on what type of data the developers using the framework want to visualize at the specific moment.

Chapter 4

Implementation

4.1 System Architecture

The system architecture is divided into three main components, each essential for the whole system to function, as seen in Figure 4.1. The system architecture works as three main layers each stacked upon the work of the previous layer and in this section each different layer in the system architecture will be explained. The high-level overview of the architecture is as follows: The whole system is built upon the data initially gathered which is the bottom layer of the application. The data gathering will be further explained in Section 4.3.

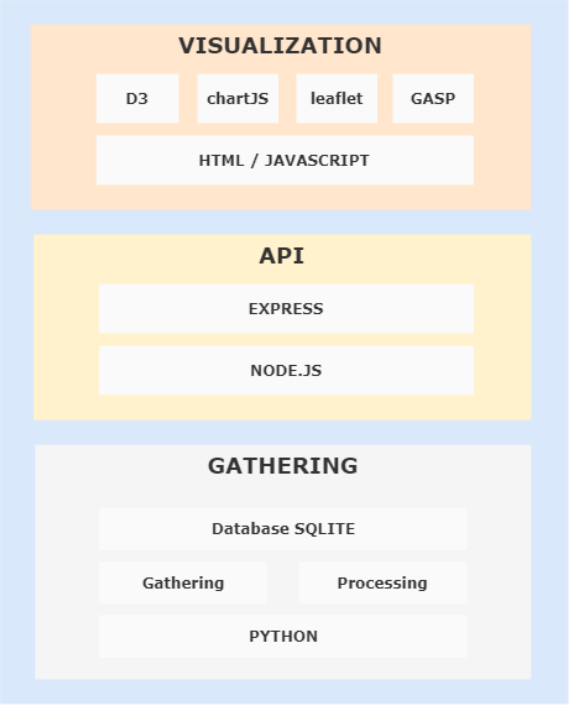


Figure 4.1: An illustration of the system architecture and how the different layers are built upon each other

On top of the data gathering layer is the middle Application Programming Interface (API) layer. The API connects the database, which is created in the bottom layer, to the user interface that is built in the top layer. The API allows the visualizations to request data. The API is built using express.js on top of a NODE.js JavaScript engine. This will be further explained in Section 4.5.

At last, on top of the API middle layer is the visualization layer. The application is built using HTML, CSS and JavaScript and the visualizations are built using the JavaScript libraries D3.js, Leaflet, GSAP and Chart.js. How we utilized these libraries will also be further elaborated on in Section 4.6.

4.2 The Data Pipeline

In order to gather the data needed to create the visualizations, we produced a pipeline consisting of 2 stages as seen in Figure 4.2. In the gathering-stage, we accumulate all the raw data available online to a local repository. The processing-stage is where we join all the raw data into a single database with different tables, and a good structure such as the Third Normal Form (3NF) which reduces data redundancy (Elmasri and Navathe, 2015). The need for this pipeline arises from a desire for automation, meaning that with the least amount of work possible the pipeline should be able to handle the process of gathering new information each year.

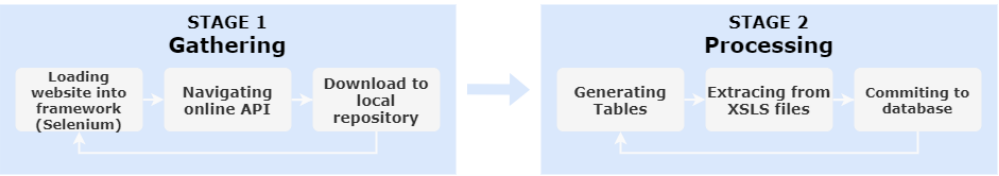


Figure 4.2: An illustration of the data gathering and its two stages of creating a useful database.

4.3 Data Gathering

Data gathering is the act of retrieving the data that is needed for the visualization. The main challenge is two-fold: One part is locating the data. Finding out if the data exists, where it is and which rights the data is protected under. The other part is how to gather the data. The book Data Visualization with Python and JavaScript, (Dale, 2016), introduces three ways of gathering the data needed in order to create visualizations. The three ways are:

- Retrieving the data as a raw file using an HTTP request.
- Use an already made API that enables developers to request data from a database.
- Scrape the data using a web scraper such as Selenium or Scrapy.

To create this specific visualization application, a combination of these methods had to be made. This combination will be described in Section 4.3.2.

4.3.1 Where to Find the Data

Research showed that many of the official documents by the government involving education and research are offered by the same site, namely: uddannelsesstatistik.dk which translates to "education-statistics.dk". It turns out that all public schools are encouraged to collect and report a large variety of statistics to the government every year. The database is then offered to the public on the site uddannelsesstatistik.dk. There are two access points: a visualization tool, which is rather slow, as well as an API that offers extraction from the database in the form of .XLSX (excel) files. This made the data-gathering stage much easier except for two issues: The first is that there was a

lot of duplicated data and second issue is that once the files leave the website, the link to uddannelsesstatistik.dk's database is broken, making the file a static output. This mean that all manipulation with the file has to be done in the online excel sheet.

4.3.2 How to Gather the Data

As described in previous section, the main challenge with retrieving the data and gaining full control over it, is that the excel sheets becomes static files once they have been downloaded. We created a workaround to this problem by using a combination of the three methods from Section 4.3.
Since the access to uddannelsesstatistik.dk's database is closed after the excel sheets are downloaded, every sheet had to be fully expanded and manipulated in order to retrieve all of the information within them - all before downloading the file. To achieve that a python framework, Selenium, was used.

Selenium

Selenium is a web automation framework that enables developers to gain full browser capabilities. This feature makes it a favorable framework to use, as compared to Scrapy that does not allow for DOM-manipulation (Gundecha, 2014). Selenium made it possible to create a script that automatically navigates to the individual excel sheet and then performs a list of tasks such as pressing specific buttons, expanding lists, checking boxes until the document is fully exposed and all information is available.

4.3.3 Gathering Stage

There are pros and cons to using selenium this way. The advantages are that the government keeps updating the same sheets which means that the document manipulation of each different sheet from uddannelsesstatistik.dk only has to be coded once. However, this means that the manipulation of every excel sheet has to be hard-coded manually. We approached this challenge by developing a software layer on top of selenium that can be fed a set of instructions, interpret them, and execute the given commands. We then produced an instruction set for each excel sheet as seen in Figure 4.3. After the instructions have been executed on the online excel sheet, the files are downloaded into a local repository as seen on Figure 4.4
The advantage of this approach is that we do not have to program each specific script. If the government releases completely new data sets, we can simply create a new instruction-file and fill it with the instructions necessary to gather the data. Another advantage is maintainability. If a sheet changes a value, adds a new column or changes names of data points we do not need to go into the code and change it. We can simply change the instruction file to accommodate the new changes.
The fetching software is written such that it supports an array of commands, which can be seen in Appendix B. These are created with high cohesion meaning that new commands can be added if needed without affecting already created commands, which increases maintainability.

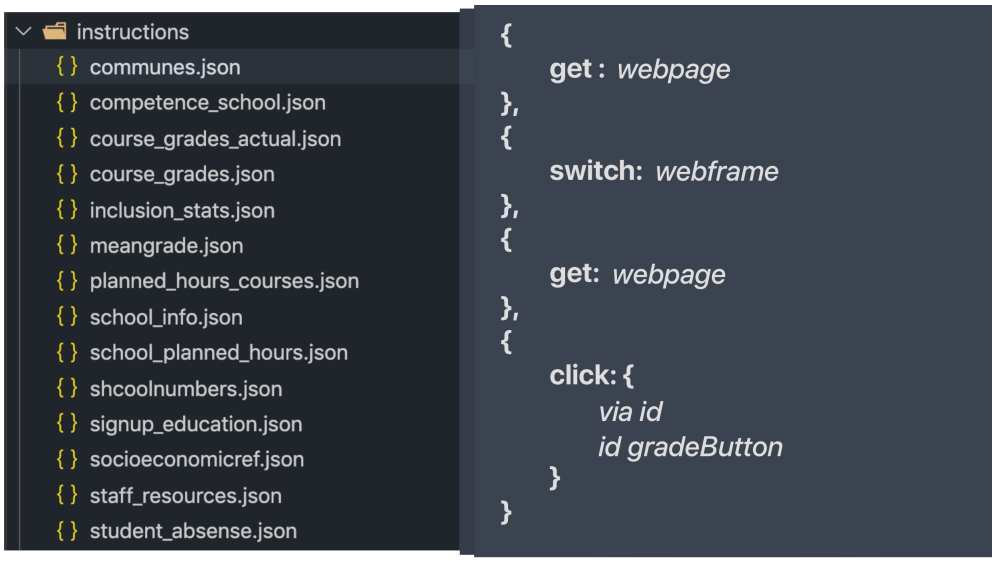


Figure 4.3: Illustration of the instruction-files with a depiction of pseudo instructions

Figure 4.4: Illustration depicting the fetching software's process. As n instruction set is loaded in, the software interprets it and acts accordingly. Once the execution is done, an excel sheet is downloaded

4.3.4 Processing Stage

Now the processing of the raw data and insertion into our database may begin. An extraction script for each, now local, excel sheet has now been created. The extraction script's role is to load the data from the excel sheet into a data frame, creating the relevant database tables and then locating the relevant rows/columns to where it should begin extraction from the data frame to the database. For this project, the database schema chosen was the 3NF. The argumentation for the schema choice is that 3NF ensure that there is minimal duplication of data. This property is good as it keeps the size of the database to a minimum, thus increasing efficiency when querying. Which in the end amounts to better user experience as the whole software will run more efficient. The extraction is done with the Python framework Pandas. Pandas enables us to directly read rows and columns from an excel file.
Each excel file has been manipulated in a way using selenium as described in the previous section, such that every row contains the school's name, commune, year and the data reported by a given school. This means extraction is simply a task of running through the file row by row and inserting the relevant data into the table(s) in the database as seen in Figure 4.5.

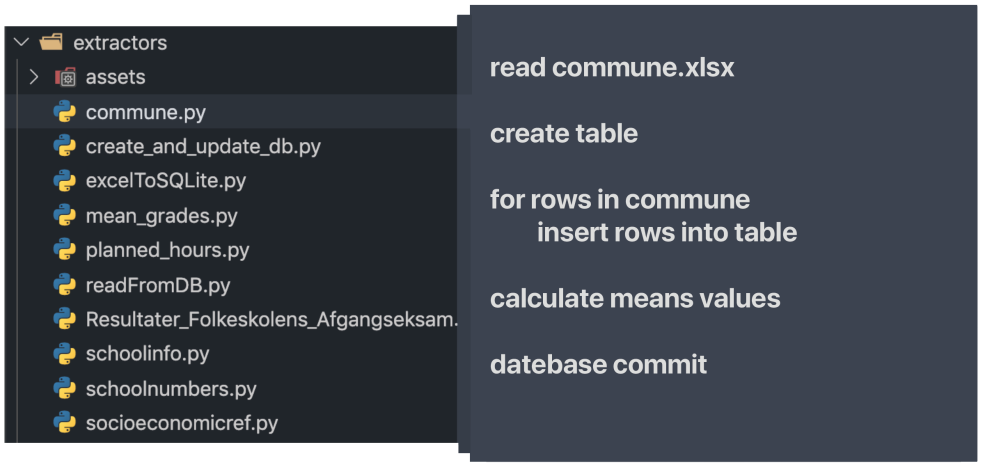


Figure 4.5: Illustration of the extraction script with accompanying high-level pseudo code.

4.3.5 Database Choice

We chose to use an SQLite database for this project since the queries are closely related to that of a normal SQL database whilst allowing us to store a single database file. This approach is viable for our project as we can then use the database directly as it is embedded into the end program or we can choose to create a client-server engine to serve data (Owens, 2006). These examples can be seen on Figure 4.6. The advantage of this approach is that we in the processing stage, where we generate and commit to the database tables, as seen in Figure 4.2, are able to interface directly with the database. When our visualization application then needs the data, our API software is able to embed the database file and set up a client-server engine with Express.js where it on request serves the data. It is preferable to create the API as a client-server engine since it allows the API to serve just the data needed for the visualization, meaning that the client does not need to store all the data locally. Another notable upside is that this structure allows for easy collaboration as the database file can be shared and used in other projects. The SQLite format is also supported by multiple different programming languages making it an ideal choice.

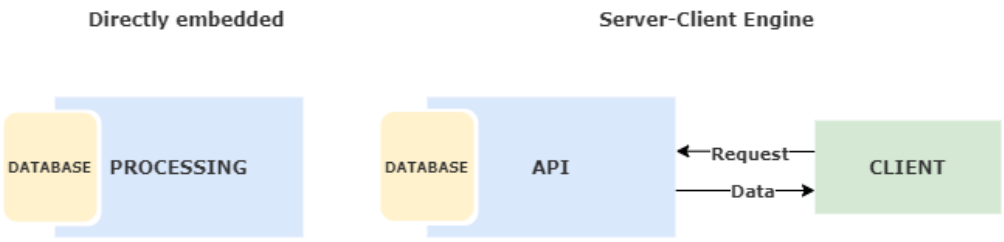


Figure 4.6: Illustration of the difference between directly embedding the database into the software and the Server-Client Approach.

4.4 Data Statistics

To give an overview of the data gathered from uddannelsesstatistik.dk, a table detailing the statistics has been made, as seen in Figure 4.7. It is important to note that the values provided in the figure (that reflects the scraped data), might deviate from what UVM.dk states, which is the official website owned by the Ministry of Children Education. The reason that the data deviates a bit is because that some separate departments of a school are technically a part of one school, which UVM.dk only counts as one school, where the data from uddannelsesstatistik.dk counts the departments as being separate schools.

Data Category	Value *as of 2019
Number of schools collected *(including 10th grade & special needs specific institutions & boarding schools)*	2449
Number of schools that did not disclose any information *(Grades, Absence,Students, Well being, Socio-economic status, Competence coverage)*Reasons unknown*	378
Number of communes	98
Earliest recorded data	2009
Total number of students (0.-10 grade)	704,573

Database information	Value
Number of unique rows in database	395,108
Number of unique columns in database	84
File size of database(kb)	15,644

Figure 4.7: An illustration of the data collected from uddannelsesstatistik.dk as well as statistics of our database

4.4.1 Data Cleaning and Data Reduction

Our data gathering does not rely on input from sensors that can give misleading or error-prone outputs that needs correcting. We do therefore not clean the data set, but must however still acknowledge that there are missing entries, e.g. schools that have not submitted data. We chose to amputate these data points, instead of using data imputation to predict them. The reasoning behind this decision is that trying to predict a schools number of students a certain year, or average exam grade could lead to a misleading output.
We have utilized Data Reduction in the form of a dynamic clustering of data points (schools) on the map in the Filtering Phase in order to avoid a large number of markers

on the map at one time, which might happen in the inner cities where there are high number of schools gathered in one area. A color is assigned to each cluster, based on the number of schools accumulated within a cluster as seen in Figure 4.9. The color scheme chosen is the YlOrBr Sequential Color scheme designed by Paul Tol, which is a tweaked version of ColorBrewers YlOrBr scheme. Sequential color schemes are used when data is ordered from low to high (Tol, 2019). The ranges defined as seen in Figure 4.8 ensures that the color coding of the cluster are consistent with any degree of zoom. It is important to note that we have chosen not to normalize the data by color coding based on population density in a commune or in other ways to normalize the clusters. Not normalizing clusters is essentially just indicating where larger cities are located, however the data we are representing is a sequential scale based on volume and not schools pr. citizen, so it is accurate that there are more schools in the most populated areas.

YlOrBr Color									
Schools within a cluster	<5	<15	<25	<50	<75	<100	<200	<300	300+

Figure 4.8: An illustration of how the color of the cluster reflects the number of schools within it.

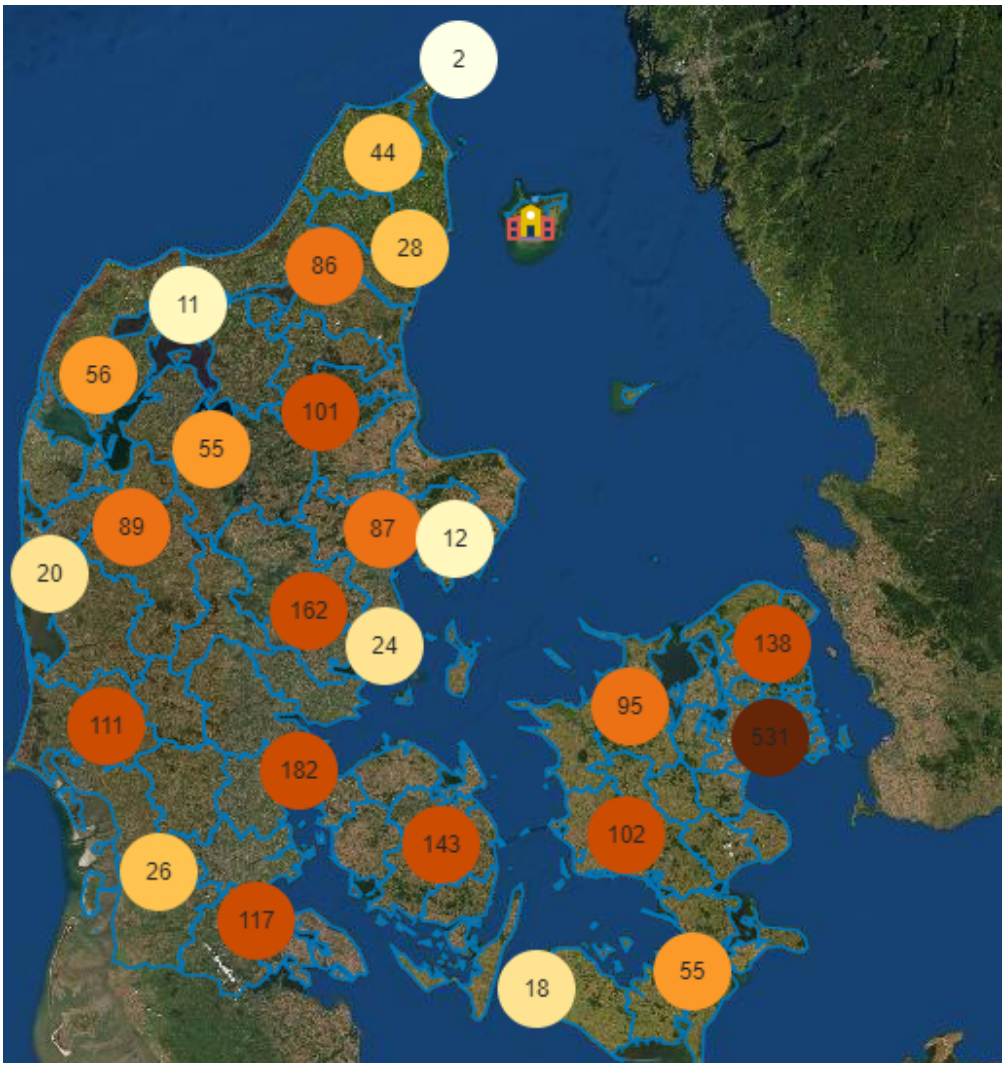


Figure 4.9: A view of how the markers are aggregated into a cluster to avoid overload of markers on the map. A marker without a cluster can be seen in the top of the figure.

4.5 API

The database full of school data is now ready to be served to the client and we need a piece of software that can handle the connection between them and transform the result to a format compatible with our visualization tool.
When building an API there are a lot different options as to which framework to use. Python with Flask offers fast and flexible development. NODE.js with Express offers a wide array of extensions and are cable of handling large input/output (I/O) scenarios (Lei et al., 2014). We chose to build the API with NODE.js and Express as we presume that a large number of users will request data resulting in IO-intensive situations. Express will thus enable us to create the best user conditions as it will allow for faster data delivery to the end-user as concluded by Lei et al. (2014).

The API's job, as seen in Figure 4.10, is to handle the data transfers between the

user and the database. When data is needed to create a visualization, a GET-request is sent to the API. The API runs the query specified by the request and returns the answer to the client in the form of a JSON file. When a user enables a filter, another GET-request is sent which then again prompts a query in the database by the API which then returns a list of all of the relevant statistics. The browser will then return the statistics that are within this filter. E.g when a user wants to filter by mean grade, the application sends a GET-request to the API asking for all of the mean grade records. The API returns them and the browser applies the given filters and displays the results.

Figure 4.10: An illustration of the API's data transferring process.

4.6 Building the User Interface

The following section will explain how we implemented the user interface by following the DEEVA Framework introduced in the previous chapter. The core of the DEEVA Framework is the taxonomic division of the application into five main phases. This section will mainly focus on how we implement these five phases to build a strong basis for the user experience and to test our hypothesis.

4.6.1 Overview Phase

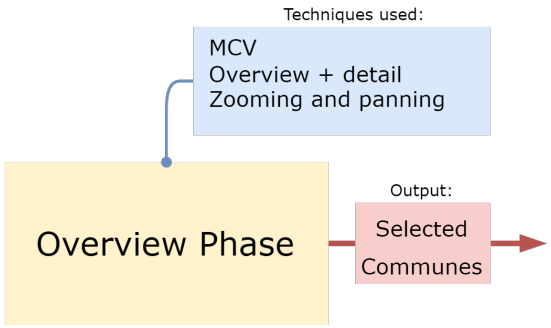


Figure 4.11: A highlight of what the Overview Phase outputs and serves to the upcoming phase along with the techniques used within.

To implement the overview phase we need to make sure of which visualization techniques are compatible with the specific phase. Since the data in our project is based on public schools, we chose to focus on geography as the main element in the overview. The reasoning behind this is that using a map provides an extra element to the overview. Instead of merely looking through a complete list of schools, the schools are mapped directly to their address on the map making the schools more tangible as they are now visualized. Another effect is that it is also possible to see the density of the schools which otherwise would be hard to visualize without a map. Having identified what to highlight, we now have to decide what we need to implement the phase. An Illustration of the output of the Overview Phase and the techniques used within can be seen in Figure 4.11.

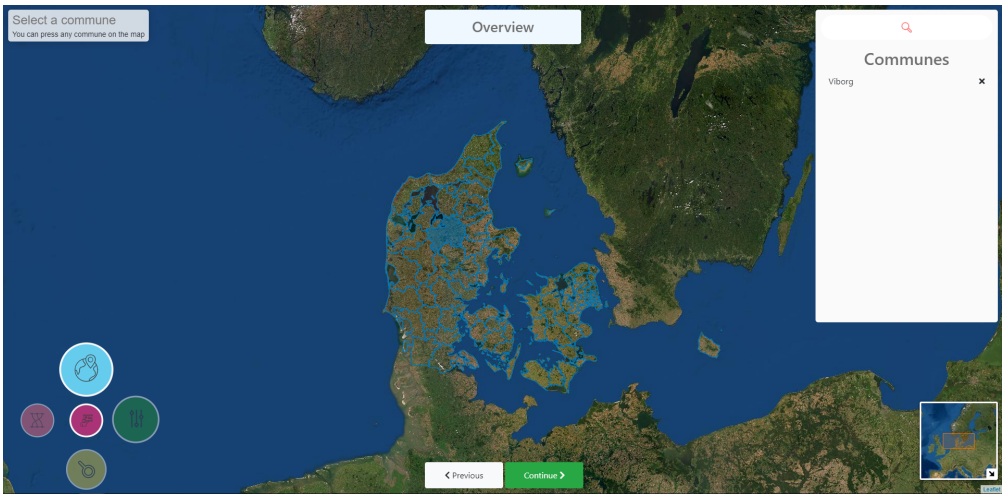


Figure 4.12: The overview phase is built with five main components, the navigation wheel in the bottom left, allowing users to both navigate and maintain overview of current context(phase), the commune indicator in the top left showing what commune is being hovered over. A mini-map in the bottom right corner helping reduce memory load when zoomed in on the map. A commune list located on the right that automatically adds each selected commune. Lastly a search bar integrated with an auto complete function is located wihtin the list container.

Map and Mini-map as Multiple Coordinated Views

One way to implement a location-based overview is with a map as seen in Figure 4.12. We divided the entire country into communes that the user can click and highlight based on an integrated GEOJSON file that outlines each commune within the database as seen in Figure 4.14. We decided to use multiple coordinated views consisting of a map and a mini-map so the user can maintain a mental map while exploring. This provides an Overview + Detail aspect. The user should be able to see a detailed view of the map on the main screen, while a small coordinated view provides the overview in the form of a mini-map as seen in Figure 4.13. If the user wants to view a different location it may either be done by dragging the main view across the screen or point to it using the mini-map and then be panned over to view it on the main screen.

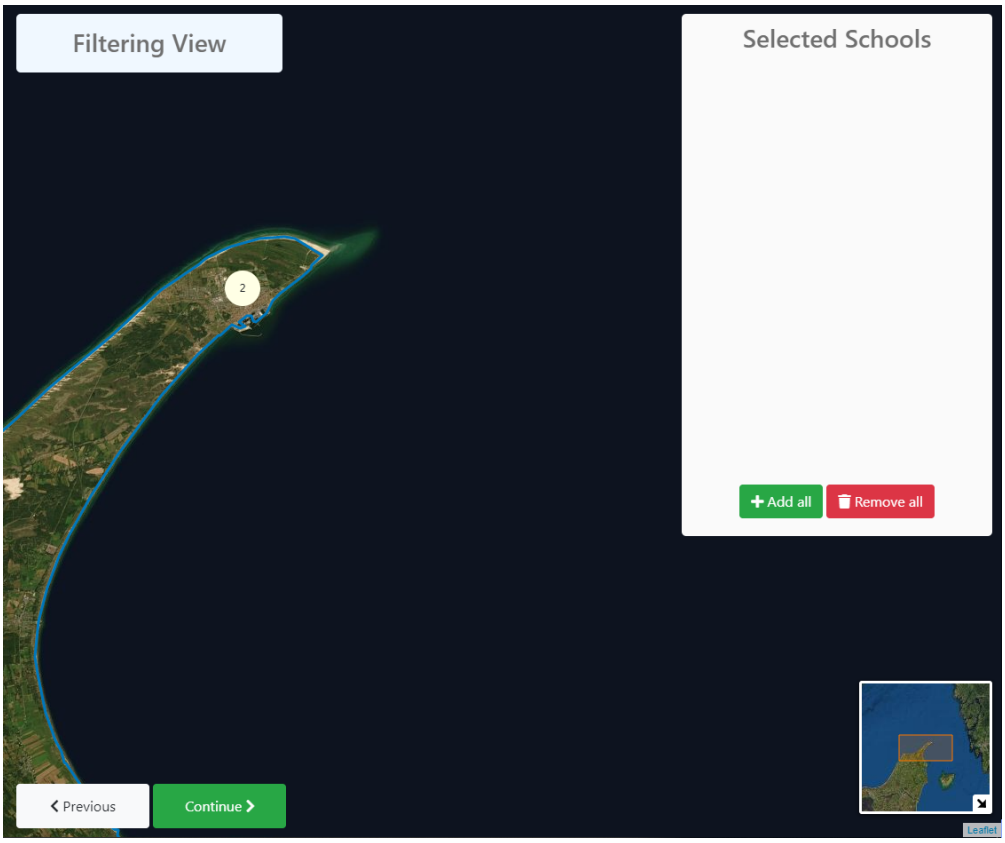


Figure 4.13: A screenshot of the coordination between the multiple views.

Building the Map with Leaflet.js

Leaflet is an open-source JavaScript library for interactive maps. Leaflet is focused on simplicity, usability, and performance and is, compared to other libraries such as MapBox.js or Google Maps, very light-weight (Crickard III, 2014). This is the main reason the specific library was chosen for our Map visualization. Keeping the JavaScript libraries light-weight means that less code is being loaded into the application which therefore affords more processing power elsewhere. Leaflet is also a flexible library that functions across all major desktop and mobile platforms which guarantees that the map will work with any device the user wishes to use. This is especially important given the large variety of users that will use the application. Figure 4.14 displays Leaflet incorporated in the system.

Figure 4.14: Screenshot of the leaflet API combined with GEOJSON that outlines the selected communes.

Zooming and Panning

When switching between two different communes, the user will experience a zoom and pan between the two. This will enable users to maintain focus, as described in the related work section with the article Smooth And Efficient Zooming And Panning by van Wijk and Nuij (2003).

Auto Completing Search Functionality

Based on the paper from (Wattenberg and Kriss, 2006), we decided to implement an auto completing search function that reacts to each keystroke the user enters. This has the potential to nudge some users into further exploration, because it might suggest search results that the user had not initially thought of. For users with specific communes in mind, they have the ability to use this feature as a regular search bar and find the commune they seek since they only need to enter a few letters before the commune shows up in the list of suggestions as seen in Figure 4.15.
Users should also be able to search by a specific school, since some of the user requirements from Chapter 3 highlighted the need to view single schools. This can be done by adding the ability to search for specific schools to the existing search bar and then highlighting the commune where the school lies within.

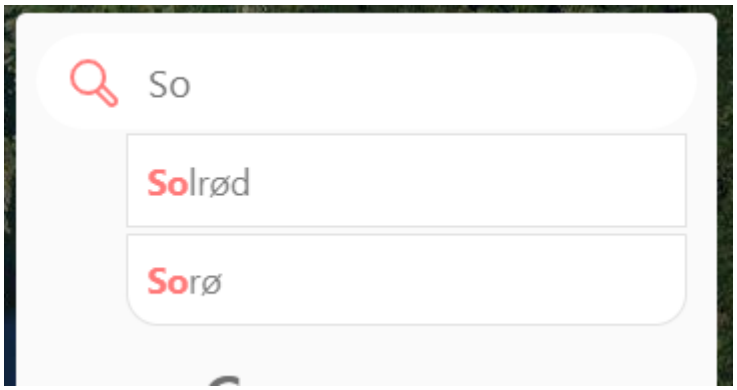


Figure 4.15: A view of suggested communes that appear when users enters letters into the search bar.

Reducing Short Term Memory Load

Reducing short term memory load is not a concept from data visualization, but it is still something we applied in the overview phase, because it enables users to stay focused on their explorations. To reduce the users short term memory load, we created a list where the selected communes are added to as seen in Figure 4.16. The list highlights the communes which schools will be transferred to the filtering phase.

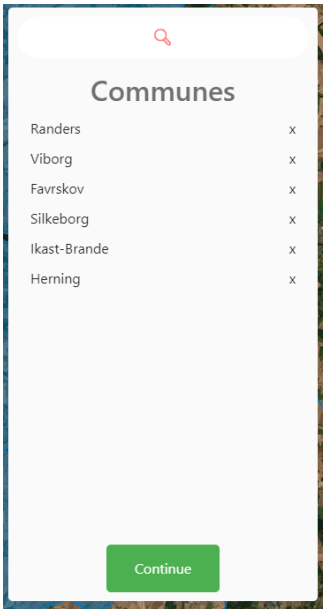


Figure 4.16: A screenshot of the list that contains all of the selected communes in the Overview Phase.

4.6.2 Filtering Phase

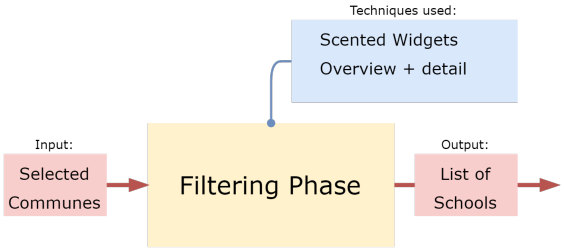


Figure 4.17: An illustration of what the Filter Phase takes as input from the previous phase, the outputs it serves to the upcoming phase along with the techniques used within the phase.

Implementing the filtering phase is very much dependent on what data we have. Our data set has a lot of different facets. The filters chosen for our implementation are based on the most requested information by the users in the questionnaire in Chapter 3. The filtering phase is introduced when users have selected a single or multiple commune(s) that they wish to explore further, as seen in Figure 4.14. This is presented as a side-panel where the users can filter based on the statistics available such as average grades, student absence or student well being. The following section will explain how and which technologies used to implement the Filtering Phase. An Illustration of the I/O of the Filtering Phase and the techniques used within can be seen in Figure 4.17.

Figure 4.18: A screenshot showing how markers are placed on the map giving details about each school.

Overview + Detail

Once a single or multiple communes have been selected, a zoomed view of the commune(s) is presented where each school within that selected commune is placed on the map with an interactive marker see Figure 4.18. Clicking on the marker brings up a pop-up box displaying general information about the given school such as school name, address, phone number etc.

Scented Widgets

To display additional information and promote exploration, we decided to use scented widgets to adjust filters. This is done by using the JavaScript library D3.js. We chose to use D3 as it allows for great customization of visualizations. This customizability enables the creation of unique visualizations such as Scented Widgets. (Zhu, 2013) We chose to use social scents, which means that the scent above the widgets contains information gathered from previous users filters and will over time display the most searched parameters. These scents will decrease the mental load of retrieving information and improve exploration of the data (Willett et al., 2007), by allowing the user to view where important discoveries are, assuming that users gravitate toward the same patterns over time, and where there is unexplored territory within the data set as seen in Figure 4.19.

Storing the data behind the scented widgets is handled by the API. Once a user enters the Filtering Phase, a GET-request is sent to the API returning all the scent data. When the data is received by the client the widgets are built by appending several rectangles on top of the widgets where the height of the rectangles is dependent on the value of the data fetched from the API. Because we chose to use a social scent, we also need to store the filtering parameters applied by the users. This is done by sending the parameters to the API via a POST-request once the user enters the next phase of the framework which is the Detail Phase. When the API receives the POST-request it calculates how the next scent should look and stores the data in the database.

We also ensure that all the data collected is only gathered as values used by the scented widgets and is completely anonymous. The users will be informed of this to resolve any possible concerns regarding data privacy they might have. It is also important to note that users emotional response to a platform collecting personal information might alter their exploration habits because of the sensation of feeling monitored, (Zuboff, 2016).

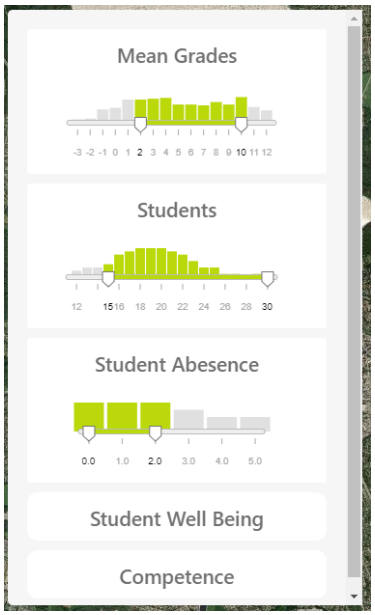


Figure 4.19: An illustration of how the scented widgets provide a social and exploratory aspect to the filtering process by showing more common and uncommon searches by other users.

Offer Informative Feedback

Offering immediate and informative feedback was implemented through dynamic queries to ensure the ability for the users to immediately realize the results of the filtering. This is done by the filtering phase rendering the results dynamically by both removing the markers on the map that do not fit within the bounds of the filter and simultaneously adding the schools that fulfill the filter in a list on the side. This allows the users to increase the speed of their exploration (Card et al., 1999).

4.6.3 Detail Phase

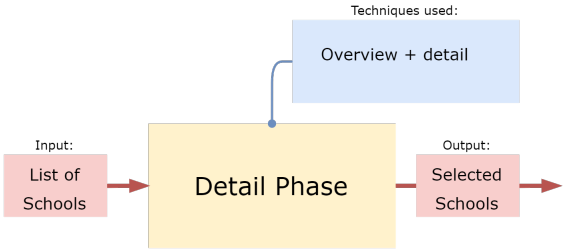


Figure 4.20: An Illustration of what the Detail Phase takes as input from the previous phase, the output it serves to the upcoming phase along with the techniques used within that phase.

The next phase we implemented is the Detail Phase as seen in Figure 4.21. The Details Phase presents the users with a new window that overlays the map from the previous phases. On the left screen, all schools that has been chosen through the Filtering Phase are available. The Detail Phase almost works as an intersection between users whose goal is just to lookup information about a single school and users who are trying to explore the data by moving on to the relation phase. An Illustration of the I/O of the Detail Phase and the techniques used within can be seen in Figure 4.20.

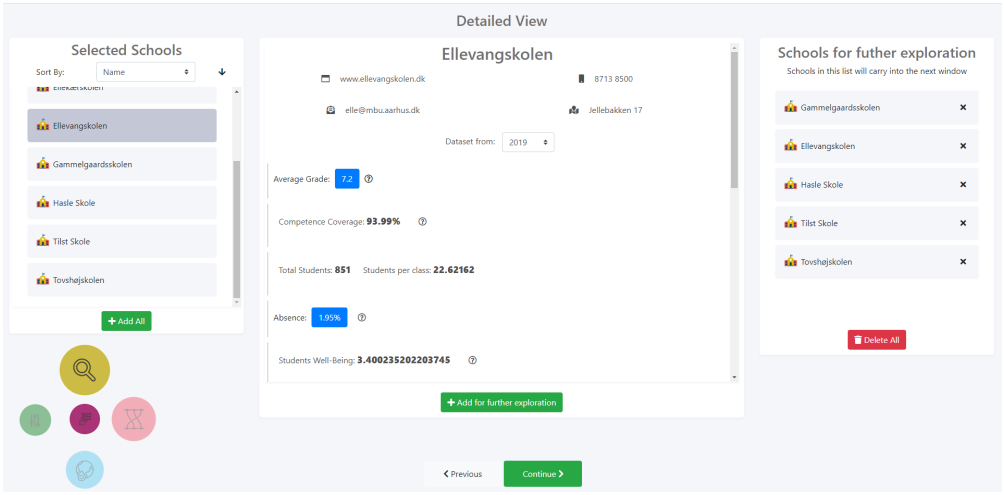


Figure 4.21: An illustration of how the Detail Phase is structured. A list of selected schools from the previous phase is illustrated on the left, where details about each school with expandable information boxes in the middle, and lastly all the added schools ready for the next phase on the right.

Overview + Detail

In the detail phase, we used Overview + Detail in a way where users can click on a specific school, which then shows all of the relevant statistics for the given school. We did it this way because some users might want to just explore a single school and do not wish to put the school into the context of other schools which is done in the Relation Phase. Users are still able to add the schools that they want to further explore into a secondary list. All of the schools that has been placed within this list are used in the upcoming Relation Phase.

We also implemented a feature that allows the user to hover over a question mark beside the data to get a comprehensive explanation of the values as seen in Figure 4.22. This ensures that the novice user has the ability to better comprehend the information. This feature is also an example of Overview + Detail.

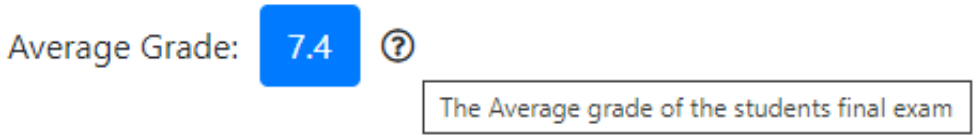


Figure 4.22: A detailed explanation for each data category presented for the novice user.

4.6.4 Relation Phase

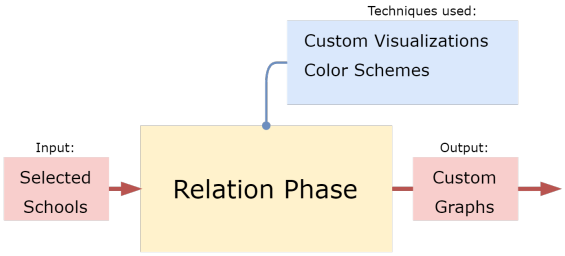


Figure 4.23: An illustration of what the Relation Phase takes as input from the previous phase, the output it serves to the upcoming phase along with the techniques used within that phase.

As described earlier in Chapter 3, the Relation Phase is where users are able to compare schools with each other. After the Detail Phase, users should have a complete list of all schools that they wish to explore further.
The Relation Phase is centered around a main graphing area where users can choose between different categories on the x- and y-axis through a drop down menu as seen in Figure 4.24. This means that if users want to see a graph of grades over income, they have to select them on the different axis. Allowing the users to chose what data is plotted is a better way to drive exploration, because we are not limiting the visualizations to comparisons that we as developers can think of - i.e if users want to explore a certain correlation in the data set they will have the ability to do so, even if we as developers have not thought of the specific comparison beforehand. An Illustration of the I/O of the Relation Phase and the techniques used within can be seen in Figure 4.23.

Figure 4.24: An illustration of the selected schools using color coding according to Paul Tol's color scheme as described in Section 4.6.4. On the right side of the screen all of the saved graphs ready for extraction in the presentation phase. The graphs with user determined x and y-axis in the middle and on the left all the schools selected to display on the graph.

Creating graphs with Chart.js

To create the graphs seen in Figure 4.24, we chose to use the JavaScript library Chart.js which is a high-level visualization library for rendering charts (Da Rocha, 2019). The reasoning behind this choice is that the Chart.js library presents us with an array of common graphs such as scatter plots, line charts, pie charts etc. out of the box. These are all charts that most users are accustomed to. D3.js strengths lie in the ability to create customizable visualizations. The cost of this customizability is complexity (Zhu, 2013). This is were a high-level library such as Chart.js shines through as it allows us to create common graphs with a few lines of code. Meaning we can achieve the same result with a less amount of labor.

Color Scheme

Since we are working with different schools and therefore categorical data, we decided to use Paul Tol's color scheme for qualitative data as seen in Figure 4.24. We chose the color scheme *Bright* since the number of categories (i.e schools) to visualize might vary and the *Bright* color scheme supports a number of categories compared to the *High contrast* scheme that only works well with fewer categories, (Tol, 2019). The color scheme works well since the colors are easy to distinguish even for color-blind people and as well as in a black and white scenario e.g. If you were to print it out and display the graph in a monochromatic display, all of the schools would still be recognizable from one another.

4.6.5 Presentation Phase

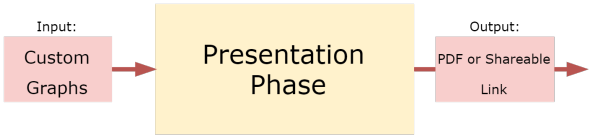


Figure 4.25: A highlight of what the Presentation Phase takes as input from the previous phase, the output it serves when the user ready to extract from the application.

The last phase we implemented was the Presentation Phase. The requirements for the Presentation Phase are that users must be able to save, share, and turn their charts into presentable visualizations. The Presentation Phase is available after a user has named and saved a visualization from the Relation Phase and clicked on continue as seen in Figure 4.26.

The Presentation Phase is built as a new window where all of the previously saved graphs from the Relation Phase are displayed and accessible on the left. In this window users are able to open their previous visualizations and create a title for them as well as annotations. The requirement of sharing findings is fulfilled by having a simple button on the right side of the screen, where the user can get a link that provides access to the chosen visualizations. Users are also able to download the currently highlighted visualizations. To do this, the user simply clicks *Download PDF* which will initialize the download. An Illustration of the Input from the previous phase and what the Presentation Phase outputs as well as the techniques used within can be seen in Figure 4.25.

Figure 4.26: An Illustration of the Presentation Phase. On the left are the two previously saved visualizations, the middle shows the graphs created and the ability to annotate. On the right the users are presented with a set of options to extract, download or share their findings.

4.6.6 Navigating Through the Different Phases

Navigation through the different phases needs to support the framework's flow of action as well as highlight how the phases directly relate to each other. This is done by creating an animated transition between the phases and an animated circular navigation wheel seen in Figure 4.27.

(a) An illustration of the difference in opacity between the phases in the navigation wheel.

(b) An illustration of the difference in size determined by the users current phase.

(c) An illustration of how the Presentation Phase breaks the circular flow

Figure 4.27: Illustration of how the navigation wheel transforms itself throughout the use of the application. .

Transitioning between Screens

The transitioning of the screens and the animation created for the navigation wheel is synchronous and follow the same direction. The direction of the transitions and animation follows a right-to-left direction. The right-to-left animation indicates to the

user that they are going forward in the application. The opposite applies for going backwards in the application.

Implementing a way to visualize the current and upcoming phases on the navigation wheel was done through scaling. The largest button placed at the top represents the current phase, while the next phase's size is decreased by 20%, as illustrated in Figure 4.27a, this continues for the other phases as well.

Creating the Animations

When animating objects or SVG files within a web-page, animation libraries are especially useful as they cut down on development time and help simplify the otherwise difficult process of animation. We chose to use GSAP(Greensock Animation Platform) for our animations. GSAP is a fast performing, light-weight JavaScript library (Shapiro, 2015), which is one of the reasons we chose it. We are working with visualizations that can be interacted with which means that the animation should run as smoothly as possible to increase the inherent understanding of the motions. Keeping JavaScript lightweight benefits all users of the system as performance increases. Making the system run smoothly on low-end computers as well as on high-end. GSAP is also compatible with all modern web-browsers. This ensures that all animations included within the platform are accessible for all users no matter what device they use. This is especially important given the large variety of users that will potentially use the platform.

Defining Click-ability

Each button except for the button representing the current phase have their opacity lowered, as illustrated in Figure 4.27b. When users hover their mouse over any phase, the opacity will change to full brightness and afford to the user that it is clickable.

Breaking the Circle

An important distinction is made for the presentation phase. It was intentionally placed in the middle of the navigation wheel and therefore breaking the circle, as seen in Figure 4.27c. This emphasizes the fact that the users are entering the last phase and breaking the circular flow, as described in the DEEVA Framework. If the users are not ready to navigate into the Presentation Phase, they will have the possibility to navigate back to the Overview Phase and continue their explorations maintaining the circular flow which is key to the DEEVA Framework as seen in Figure 4.28.

Figure 4.28: The user is prompted with a message to ensure that they are ready to break the explorative cycle and transition into the last phase.

Color Choice of the Navigation Wheel

The colors chosen for the navigation wheel follow same qualitative color scheme of Paul Tol as used in Section 4.6.4. The scheme will ensure usability and accessibility for our broad user group, especially for users that might suffer from colorblindness. The navigation button can be seen in detail on Figure 4.27.

Chapter 5

Evaluation and Discussion

5.1 Evaluation

To evaluate the effectiveness of the DEEVA Framework we need to test it on potential users. These user tests will consist of qualitative data-gathering techniques based on the Cooperative Evaluation as described by (Benyon, 2014). We will focus on data that can tell us about the participants walk-through, difficulties with achieving their goals, and answers to open-ended questions. This means that data such as time spent, error rates, and other quantitative data types have been given a lower priority. The reasoning for this is that users have many different goals and we cannot argue that one specific goal is more important than the other e.g if a user wants to utilize the platform for research, we cannot evaluate what the optimal time to perform such a task is, given that it is dependent on the specific research. We do however acknowledge that time spent on a task may indicate how well a user interface has been designed, but our focus is mainly on the implementation of the DEEVA Framework and how well the user circulates through the application.

5.1.1 User Tests

Following the guidelines by the Cooperative Evaluation, a list of tasks was created. These tasks were focused on testing the implementation of the DEEVA Framework and how well the framework supports a visualization application. Each of the tasks were created to highlight a specific concern, that will be described after the presentation of the given task down below. Through the evaluation process, the participants were presented the following four tasks:

- Task 1: *Locate the public school that the participant went to and retrieve the average grade for that school.* This task highlights the users ability to move between the Overview, Filtering, and Detail Phase. The goal of this task is to gain knowledge on how the user handles the input and outputs of going through multiple phases, which is something that will happen multiple times during the use of the application.
- Task 2: *Identify the public school containing the highest average grade within the commune.* This task highlights how users will approach the need to find a

specific detail of a school. There are multiple ways for a user to complete this task, so the goal is to identify which way the participant prefers, because this might tell us how well the visibility of the different features compare to each other.

- Task 3: *Compare the average grades of all schools within the initially selected commune to all schools in any different commune.* This task tests the usability of the Relation Phase and how well users can select schools from different parts of the map to compare. We wanted to have a task that reflects the actual comparisons and how well the user can navigate the graphing screen of the Relation Phase. This is because the user requirements suggests that performing research such as identifying trends over time is an action that multiple users groups will find useful.

- Task 4: *Create one or more graphs depicting the comparisons from the earlier task and download the visualization to a PDF file.* This task makes sure that we have achieved a high degree of coverage of the application through the user tests, and the Presentation Phase would remain untested if not for this task. We want to make sure that all users are able to achieve their goals and presentation of graphs was one of the desired functionalities.

Throughout the evaluation and while the tasks were being solved, we encouraged the users to Concurrently Think Aloud (CTA) to get an understanding of the users' thought process during the use of our application (Bergstrom, 2013). By keeping the user tests in a semi-structured state, we could ask the users follow-up questions such as *"Why did you click there?"*, *"What did you imagine would happen after that action?"* and *"What do you think the system tells you to do here?"*. The combination of the Cooperative Evaluation form, the thinking aloud technique, and the follow-up questions, allowed us to understand the user's impressions of the system and how they perceived the DEEVA Framework.

5.1.2 Results

All three participants, two female and one male, in the evaluation were in the age group of 20-30 and have or are currently attending a higher education. None of the participants have children. One of the female users is currently searching for a place to purchase real estate. This means that two of the participants can be placed in the "just for exploration" category while the last participant had a clearly defined goal. It is also worth noting that the participants are technology literate and cannot be categorized as novice users when it comes to interacting with electronic devices. The following section will describe and reflect on the results from the user tests.

A common theme throughout the tests was that participants struggled with navigation, especially when they were transitioning through the different phases of the framework. A lack of affordance as described by Norman (1988) on how to proceed to the next phase meant that participants felt confused. This turns out to be one of the main aspects of the evaluation and will be discussed further in the Discussion in Section 5.2. Eventually, all participants figured out how to navigate throughout the

phases, but it wasn't as intuitive as we hoped. A consequence of not having clear distinguishing transitions between each phase meant that some participants did not comprehend why certain actions did not work as expected, e.g. a user tried selecting an additional commune (a feature of the overview phase) in the filtering phase and expressed a feeling of frustration and loss of control. Users would immediately try to access a single school in the overview phase and try to explore it individually in a way that suggests skipping the filtering phase as a whole.

An interesting discovery was made during the evaluation. All users did generally solve the different tasks, but the way they achieved the solutions varied. The application allows for multiple ways to complete the tasks, and every participant had their own way of navigating the application and finding the correct information. An example is the task of finding the school with the highest grades. This could be solved by either filtering with the scented widgets so that the map dynamically displays the school with the highest average grade or by using the *"sort by:"* feature in the detail phase and identifying the top element on the list. Lastly, the task could also be solved by plotting the schools on a chart and looking at the different schools' grades over time in the Relation Phase.

All participants found the multiple coordinated views in the overview and filtering phase useful. The users said that it provided them with a good overview of the schools and increased their exploring abilities. When asked, participants felt that it was a good way to relate data as the schools became more tangible and less abstract when placed on a map.

The participants appeared to learn the system fast. After a full run-through of all the phases, participants seemed eager to challenge the software and explore further. Nevertheless, since participants only had around 15-30 minutes to play around with the system and get familiar with it, they never had a chance of becoming more than novice users. This means that an expert user might have a different approach to the system and might be more capable of adhering to the DEEVA Frameworks circular flow of action as well as explore the features of the application differently.

5.2 Discussion

In the introduction we hypothesized that it would be possible to create a framework that developers could use to always create the best visualization product based on a heterogeneous data set for many different user groups. This is, of course, a lot to demand and we can only base the usability of the framework on the work we have done. It would be very interesting to see how different platforms such as HomeFinder and NameVoyager would implement or rework their system based on the DEEVA Framework and understand how it affects their end product.
One major concern arose during the user test in Section 5.1.2 that we want to further address: the fact that users had a hard time navigating between the different phases of the application. We have identified two possible reasons for this.

The first reason why users might have had trouble with navigating the user interface could be because we as the developers have not had the time to create better usability tests. This means that we are, at the current state of the application, unable to determine whether the outcome of the user tests is a result of the application not being user friendly enough and a symptom of not following the iterative design process enough times or the fact that the framework is not performing the way we imagined it. I.e we believe that the user interface might have been underdeveloped and therefore influenced the user tests in a way where the results do not reflect the questions we wanted answered. The second reason why users might have had issues with the navigation and felt a loss of control could be that users expected the phases to work in a specific way because the user had created a certain mental map of the application or had a different impression of what should happen at certain points in the application. E.g people were ready to explore the locations of schools right away instead of being forced to utilize the Overview Phase first. The same problem occurred where people would prefer to be directly taken to the Detail Phase after clicking on the school in the Filtering Phase. We believe that the underlying issue is that people want to use the different functionalities at different times using the application (as predicted), but users had a tendency to need different phases at different points in their exploration based on their own needs - not necessarily in the same order every time. This indicates that the circular flow of action perhaps is not the best model to base the framework upon. The workflow that the users expressed while interacting with the platform suggests that a network of phases would work better than a perfectly circular model.

Looking back at how we created the circular model in the first place, we realized that the DEEVA Framework might be based on a misinterpreted model. The circular model that we argued for in section 3.4.2 tried to express the ability for users to freely move between functionalities by referring to the flow of action of other online application such as Facebook.com, as seen in Figure 3.7. However, what we realized is, that the circular model also has an inherent linear structure like a clock always moving in the same direction, which suggests that the model that online social medias flow of action is *actually* based on, is in fact a network-like model instead, as mentioned in the section above. We have learned that this misrepresentation of the circular model has impacted the framework. Creating a second iteration of the DEEVA Framework and basing it on a network-like model as seen in Figure 5.1 gives the users the possibility to create smaller iterations of exploration within two phases of the framework. In other words, users now have the ability to utilize any of the phases whenever they need to, instead of being forced to follow the inherent linear pathway of the circular framework.

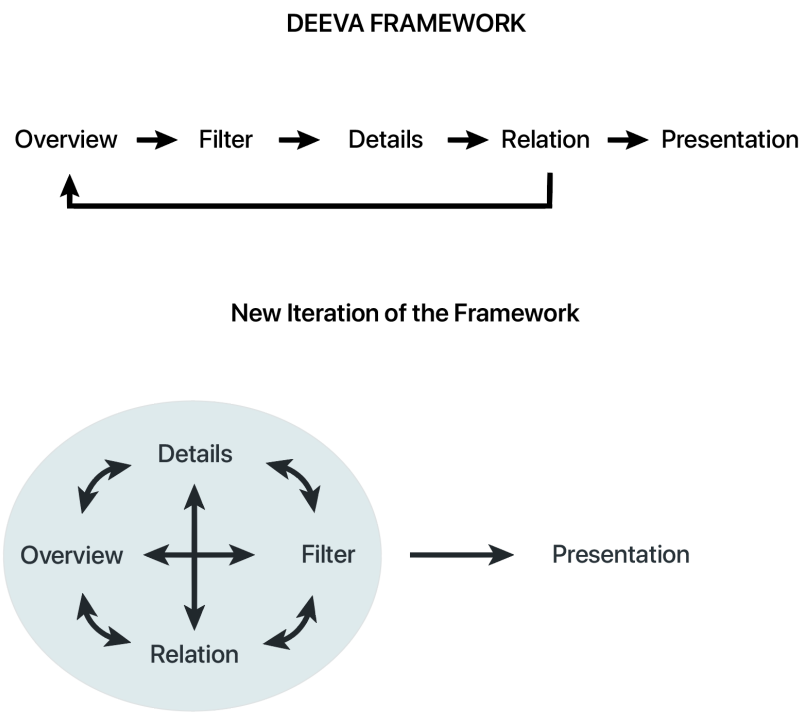


Figure 5.1: An illustration of how the phases will be placed in a network structure in the next iteration of the DEEVA Framework.

Chapter 6

Future Work

In the current iteration of our implementation, there are several limitations that may be addressed for future work. In this section, we will introduce a few of these limitations and how we or other researchers might try to handle them in the future.

The first limitation that might be addressed in a future project is the fact that novice users find the system difficult to understand as discussed in the results of the evaluation section. To solve this problem various approaches may be taken. The naive approach is to solve this problem with an interactive user guide using coach marks that helps the user through a single iteration of the DEEVA Framework. The results from the evaluations suggest that after a single iteration the users had a much better understanding of how to proceed with the exploration. This means that a guide would allow users to better grasp the features of the application.
Another approach would be to redesign the user interface to be more minimalistic and easier to comprehend the actions available. This would, of course, be a more daunting task than to implement a simple guide. It would however serve to truly show how well the DEEVA Framework performs. Another issue is that users experience increased mental load when transitioning from the Filtering Phase to the Detail Phase. The interactive map from the previous phases disappears and the selected schools transform from a graphical representation to a textual. This signals that adding a mini-map with all currently selected schools placed on it and making them visible in the Detail Phase might be a good idea.

The evaluation of the strengths and weaknesses of the framework gave insight into how well the DEEVA Framework was implemented into the application and highlighted its limitations. It is important to note that the users that participated in the first evaluation round only reflect a fraction of the future users. This means that the results extracted from the evaluation may be biased or misrepresented. To truly test the strengths and weaknesses of the framework would be to conduct an additional evaluation with a broader user group. However before conducting further evaluations, the feedback received has to be integrated into the application.

The current state of our implementation lacks more advanced interaction and visu-

alization techniques especially in the later phases of the framework. This results in the platform not fully accommodating the large number of different users. Implementing techniques discussed in Chapter 2 will help increase the ability to explore especially for advanced users and might even entice intermediate users to explore data that they otherwise would not have. Examples of advanced techniques to implement could be taking advantage of the array of different interaction modalities introduced by interactive lenses as described by (Tominski et al., 2017). Tominski et al. (2006) describes lenses such as the "Bring neighbors lens". This type of lens could be implemented into the Filtering Phase and aid users, that might have a tendency to lose their mental image of neighboring schools when zoomed in on a specific school. In addition, the relation phase could benefit from the use of the interactive lenses described by Tominski et al. (2006) where users compare schools based on graphs. Graphs can easily become cluttered especially when comparing a large number of schools. This can result in users losing their ability to explore the data based on the increased cognitive load.
An additional improvement for the relation phase would be to implement brushing. More specifically the extension of the brushing technique introduced by Koytek et al. (2017). This could improve users' ability to explore the relationships between selected data points in the plotted school graph created by the user and allow for easier comparisons between schools.

Chapter 7

Conclusion

Based on our hypothesis we developed The DEEVA Framework that categorizes the user's path throughout a visualization application with many different users as combination of a linear and a circular journey. The DEEVA Framework then lays the foundation for how to maximize the ability to explore within the five phases that the user encounters as they work their way through the application. The proposed framework was then implemented into a visualization application to test its strength and weaknesses. The evaluations confirmed our hypothesis but also highlighted that navigating between the different phases that the DEEVA Framework initially proposed did not maximize the user's ability to explore the data set. Therefore a few changes were made based on the results from the evaluation to optimize the framework. In the future, supplementary advanced data visualization techniques should be added to improve the experience for advanced users. However, the overall integration of the DEEVA Framework was successful and shows great potential to support developers in creating visualizations of heterogeneous data sets while accommodating many different user groups.

Bibliography

Christopher Ahlberg, Christopher Williamson, and Ben Shneiderman. Dynamic Queries for Information Exploration: An Implementation and Evaluation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '92, pages 619–626, New York, NY, USA, 1992. Association for Computing Machinery. ISBN 0897915135. doi: 10.1145/142750.143054. URL https://doi.org/10.1145/142750.143054.

Patrick Baudisch, Nathaniel Good, Victoria Bellotti, and Pamela Schraedley. Keeping Things in Context: A Comparative Evaluation of Focus Plus Context Screens, Overviews, and Zooming. 04 2002. doi: 10.1145/503376.503423.

David Benyon. *Designing interactive systems: A comprehensive guide to HCI, UX and interaction design*. Pearson Edinburgh, 2014.

Jennifer Romano Bergstrom. Moderating Usability Tests, 2013. URL https://www.usability.gov/get-involved/blog/2013/04/moderating-usability-tests.html#.

Stuart Card, Jock Mackinlay, and Ben Shneiderman. *Readings in Information Visualization: Using Vision To Think*. 01 1999. ISBN 978-1-55860-533-6.

Fan Chaoran and Helwig Hauser. Personalized Sketch-Based Brushing in Scatterplots. *IEEE Computer Graphics and Applications*, 39(4):28–39, July 2019. ISSN 1558-1756. doi: 10.1109/MCG.2018.2881502.

Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A Review of Overview+detail, Zooming, and Focus+context Interfaces. *ACM Comput. Surv.*, 41(1), January 2009. ISSN 0360-0300. doi: 10.1145/1456650.1456652. URL https://doi.org/10.1145/1456650.1456652.

David Coyle, James Moore, Per Ola Kristensson, Paul Fletcher, and Alan Blackwell. I did that! measuring users' experience of agency in their own actions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2025–2034, 2012.

Paul Crickard III. *Leaflet. js essentials*. Packt Publishing Ltd, 2014.

Qingguang Cui, Matthew Ward, Elke Rundensteiner, and Jing Yang. Measuring data abstraction quality in multiresolution visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):709–716, 2006.

Helder Da Rocha. *Learn Chart. js: Create interactive visualizations for the Web with Chart. js 2*. Packt Publishing Ltd, 2019.

Andre Luiz da Silva Kauer, Bianchi Serique Meiguins, Ricardo Melo Casseb do Carmo, Marcelo de Brito Garcia, and A Simões Gonçalves Meiguins. An information visualization tool with multiple coordinated views for network traffic analysis. In *2008 12th International Conference Information Visualisation*, pages 151–156. IEEE, 2008.

Kyran Dale. *Data Visualization with Python and JavaScript*. O'Reilly Media, 1005 Gravenstein Highway North, Sebastopol, CA 95472., 2016.

Alan Dix and Geoff Ellis. By Chance Enhancing Interaction with Large Data Sets through Statistical Sampling. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '02, pages 167–176, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581135378. doi: 10.1145/1556262.1556289. URL `https://doi.org/10.1145/1556262.1556289`.

Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems*. Pearson, 7th edition, 2015. ISBN 0133970779.

Niklas Elmqvist, Pierre Dragicevic, and Jean-Daniel Fekete. Rolling the Dice: Multidimensional Visual Exploration using Scatterplot Matrix Navigation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1539–1148, 2008.

Cleotilde Gonzalez. Does animation in user interfaces improve decision making? In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 27–34, 1996.

Unmesh Gundecha. *Learning Selenium Testing Tools with Python*. Packt Publishing Ltd, 2014.

Kasper Hornbaek, Benjamin B. Bederson, and Catherine Plaisant. Navigation Patterns and Usability of Zoomable User Interfaces with and without an Overview. *ACM Trans. Comput.-Hum. Interact.*, 9(4):362–389, December 2002. ISSN 1073-0516. doi: 10.1145/586081.586086. URL `https://doi.org/10.1145/586081.586086`.

Keke Chen and Ling Liu. A visual framework invites human into the clustering process. In *15th International Conference on Scientific and Statistical Database Management, 2003.*, pages 97–106, July 2003. doi: 10.1109/SSDM.2003.1214971.

Philipp Koytek, Charles Perin, Jo Vermeulen, Elisabeth André, and Sheelagh Carpendale. Mybrush: Brushing and linking with personal agency. *IEEE transactions on visualization and computer graphics*, 24(1):605–615, 2017.

Kai Lei, Yining Ma, and Zhi Tan. Performance Comparison and Evaluation of Web Development Technologies in PHP, Python, and Node.js. In *2014 IEEE 17th International Conference on Computational Science and Engineering*, pages 661–668, 2014.

Zhicheng Liu, Biye Jiang, and Jeffrey Heer. immens: Real-time visual querying of big data. *Computer Graphics Forum*, 32(3pt4):421–430, 2013. doi: 10.1111/cgf.12129. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12129.

Tamara Munzner. *Visualization Analysis and Design*. AK Peters Visualization Series. CRC Press, 2015. ISBN 9781498759717. URL https://books.google.de/books?id=NfkYCwAAQBAJ.

Glenn J. Myatt. *Making Sense of Data: A Practical Guide to Exploratory Data Analysis and Data Mining*. Wiley-Interscience, USA, 1st edition, 2006. ISBN 047007471X.

Donald A Norman. *The psychology of everyday things.* Basic books, 1988.

Mike Owens. *The definitive guide to SQLite*. Apress, 2006.

Vijayshankar Raman and Joseph Hellerstein. Potter's Wheel: An Interactive Framework for Data Cleaning and Transformation. 11 1999.

Jonathan C. Roberts. State of the Art: Coordinated Multiple Views in Exploratory Visualization. In *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization (CMV 2007)*, pages 61–71, July 2007. doi: 10.1109/CMV.2007.20.

Jonathan C. Roberts and Michael AE. Wright. Towards ubiquitous brushing for information visualization. In *Tenth International Conference on Information Visualisation (IV'06)*, pages 151–156. IEEE, 2006.

Donald B Rubin. *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons, 2004.

Joseph L Schafer. Multiple imputation: a primer. *Statistical Methods in Medical Research*, 8(1):3–15, 1999.

Julian Shapiro. *Web Animation using JavaScript: Develop & Design*. Peachpit Press, 2015.

Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE symposium on visual languages*, pages 336–343. IEEE, 1996.

Paul Tol. Colour Schemes and templates, 2019. URL https://personal.sron.nl/~pault/.

C. Tominski, S. Gladisch, U. Kister, R. Dachselt, and H. Schumann. Interactive Lenses for Visualization: An Extended Survey. *Computer Graphics Forum*, 36(6):173–200, 2017. doi: 10.1111/cgf.12871. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12871.

Christian Tominski, James Abello, Frank van Ham, and Heidrun Schumann. Fisheye Tree Views and Lenses for Graph Visualization. *Tenth International Conference on Information Visualisation (IV'06)*, pages 17–24, 2006.

Jarke J. van Wijk and Wim A.A. Nuij. Smooth and efficient zooming and panning. In *IEEE Symposium on Information Visualization 2003 (IEEE Cat. No.03TH8714)*, pages 15–23, Oct 2003. doi: 10.1109/INFVIS.2003.1249004.

Zana Vosough, Rainer Groh, and Hans-Jörg Schulz. On Establishing Visualization Requirements: A Case Study in Product Costing. In Barbora Kozlikova, Tobias Schreck, and Thomas Wischgoll, editors, *EuroVis 2017 - Short Papers*. The Eurographics Association, 2017. ISBN 978-3-03868-043-7. doi: 10.2312/eurovisshort.20171140.

Matthew Ward, Georges Grinstein, and Daniel Keim. *Interactive Data Visualization: Foundations, Techniques, and Applications*. A. K. Peters, Ltd., USA, 2010. ISBN 1568814739.

Martin Wattenberg and Jesse Kriss. Designing for social data analysis. *IEEE transactions on visualization and computer graphics*, 12(4):549–557, 2006.

Chris Weaver. Cross-Filtered Views for Multidimensional Visual Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):192–204, March 2010. ISSN 2160-9306. doi: 10.1109/TVCG.2009.94.

Di Weng, Heming Zhu, Jie Bao, Yu Zheng, and Yingcai Wu. HomeFinder Revisited: Finding Ideal Homes with Reachability-Centric Multi-Criteria Decision Making. 2018. doi: 10.1145/3173574.3173821. URL `https://doi.org/10.1145/3173574.3173821`.

Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1129–1136, 2007.

YoungSeok Yoon and Brad A Myers. Semantic zooming of code change history. In *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 95–99, Oct 2015. doi: 10.1109/VLHCC.2015.7357203.

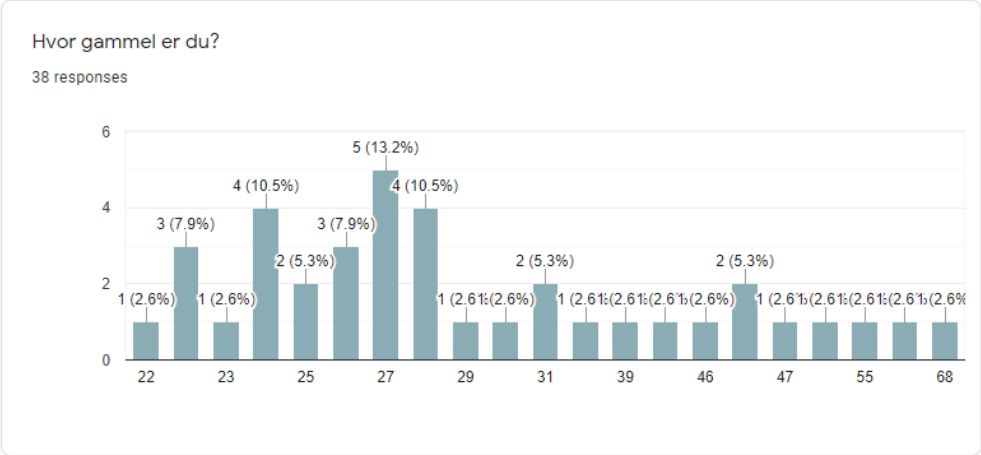
Nick Qi Zhu. *Data visualization with D3. js cookbook*. Packt Publishing Ltd, 2013.

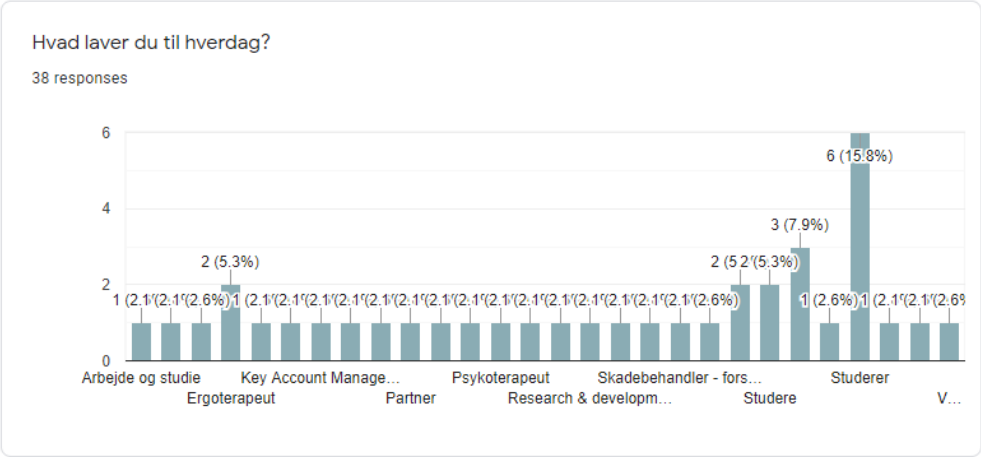
Shoshana Zuboff. Surveillance capitalism. *Frankfurter Allgemeine*, 5, 2016.

Appendices

Appendix A

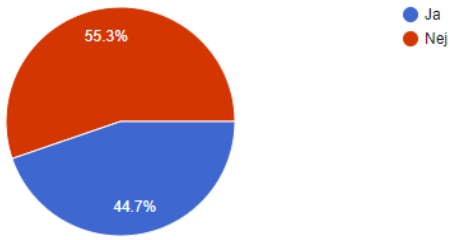
Questionnaire Answer





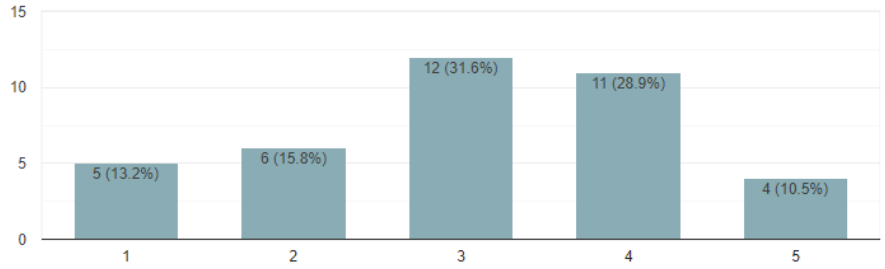
Har du børn?

38 responses



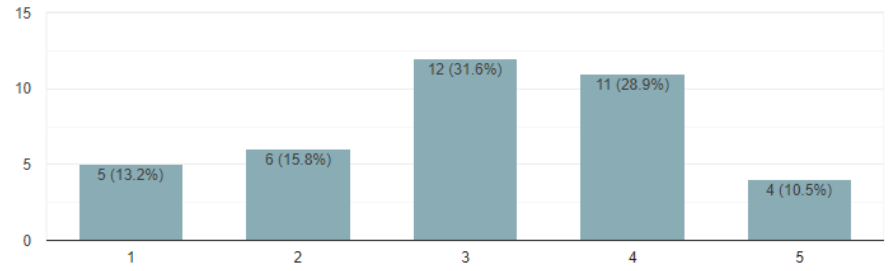
Hvor stor indflydelse har den nærliggende folkeskole på et køb af hus/lejlighed

38 responses



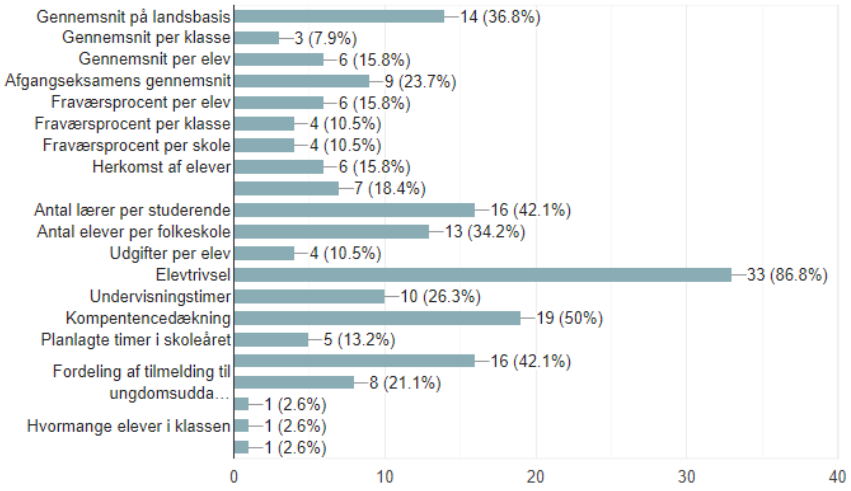
Hvor stor indflydelse har den nærliggende folkeskole på et køb af hus/lejlighed

38 responses



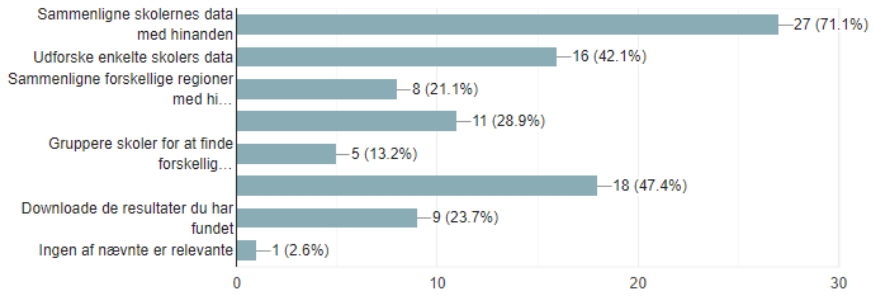
Hvilke informationer omkring folkeskoler ville du være interesseret i?

38 responses



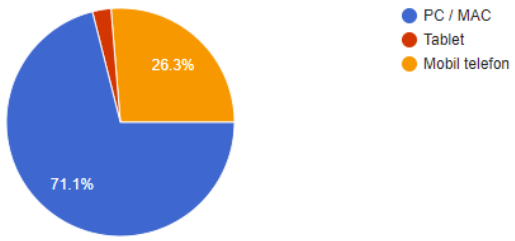
Hvilke funktioner vil du anvende, hvis du havde en platform der visualiserer folkeskolernes data?

38 responses



Hvilket device vil være din primære måde til tilgå denne applikation på nettet?

38 responses



Appendix B

Commands for Fetching Software

Extract of the Commands that can be used in Instruction files found in our documentation file on GitHub, full documentation for fetching available at: https://git.io/Jf1OQ

Instructions

Get

The get command can be used to navigate to a specified URL.

```
{
    "get": "https://sitename.com"
}
```

Wait

The Wait command can be used to stall the execution for a finite amount of time inputted

```
{
    "wait": "10"
}
```

Switch

The switch command can be used to changes the focus of selenium to an alternative window, such as an iframe window

```
{
    "switch": "WebApplicationFrame"
}
```

Click

The click command can be used to perform a mouse click on DOM objects in the window frame. The click commands can detect the DOM element by

- **id**
  The id option can be used when locating an element by the unique id identifier

```
{
    "click": {
        "find": "id",
        "value": "acceptButton"
    }
}
```

- **class**
  The class option can be used when locating an element by the class identifier

```
{
    "click": {
        "find": "class",
        "value": "roundButton"
    }
}
```

- **xpath**
  The xpath option can be used when trying to detect an element based on anything as it can be used to locate an element based on nodes in XHTML document, further detail about xpath can be read in selenium documentation here

```
{
    "click": {
        "find": "xpath",
        "value": "//select[@id='flp_srt_combobox']/option[text()='Institution']"
    }
}
```

- **dragAndDrop**
  The dragAndDrop option can be used when an element needs to be dragged to a certain position. The element is located via the text value of the DOM object. Furthermore, the dragAndDrop command has 2 additional option which is the X and Y coordinate to move the object to.

```
{
    "click": {
        "find": "dragAndDrop",
        "value": "Schoolyear",
        "x": 150,
        "y":0
    }
}
```

- **pivotTable**
  The pivotTable option can be used to click on a pivot table item in the online excel sheet.

```
{
    "click": {
        "find": "pivotTable",
        "value": "More Fields",
    }
}
```

- **pivotTableExpand**
  The pivotTableExpand option can be used to expand a pivot table item in the online excel sheet.

```
{
    "click": {
        "find": "pivotTableExpand",
        "value": "More Fields",
    }
}
```

- **pivotTableMulti**
  The pivotTableMulti can be used to click on a pivot table item in the online excel sheet where duplicate entries with no specific id are present.

```
{
    "click": {
        "find": "pivotTableMulti",
        "value": "institution",
        "listNumber": 2
    }
}
```