BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

WEB APPLICATION FOR INFORMATION DASHBOARD DESIGN

WEBOVÁ APLIKACE PRO NÁVRH INFORMAČNÍCH DASHBOARDŮ

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR — **OLENA PASTUSHENKO**
AUTOR PRÁCE

SUPERVISOR — **Ing. HYNEK JIŘÍ**
VEDOUCÍ PRÁCE

BRNO 2017

Master's Thesis Specification/18413/2016/xpastu01

Brno University of Technology - Faculty of Information Technology

Department of Information Systems Academic year 2016/2017

Master's Thesis Specification

For:	**Pastushenko Olena, Bc.**
Branch of study:	Information Systems
Title:	**Web Application for Information Dashboard Design**
Category:	User Interfaces

Instructions for project work:

1. Get acquainted with visualization tools for presenting data on a single screen - especially with *Information Dashboard*.
2. Study the principles of Information Dashboard design. Use the Information Dashboard Design book [Few].
3. Get acquainted with the principles of web application development. Perform a research of libraries which can be used for the design of charts and diagrams.
4. Design a web application which can be used to quickly prototype the information dashboards. Take the items 1 - 3 and supervisor's advices into consideration.
5. Implement the designed application.
6. Perform a usability testing, evaluate the results and propose further extensions.

Basic references:

- Few, S.: Information Dashboard Design: The Effective Visual Communication of Data. Sebastopol [MA]: O'Reilly, 2006, ISBN 978-059-6100-162.
- Johnson, J.: Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines. Burlington: Morgan Kaufmann Publishers/Elsevier, 2010, ISBN 978-0-12-375030-3.
- Rogers, Y.; Sharp, H; Preece, J.: Interaction Design: Beyond Human - Computer Interaction. 3rd Edition. New York: Wiley, 2011, ISBN 978-0-470-66576-3.

Requirements for the semestral defense:
Items 1 - 4.

Detailed formal specifications can be found at http://www.fit.vutbr.cz/info/szz/

The Master's Thesis must define its purpose, describe a current state of the art, introduce the theoretical and technical background relevant to the problems solved, and specify what parts have been used from earlier projects or have been taken over from other sources.

Each student will hand-in printed as well as electronic versions of the technical report, an electronic version of the complete program documentation, program source files, and a functional hardware prototype sample if desired. The information in electronic form will be stored on a standard non-rewritable medium (CD-R, DVD-R, etc.) in formats common at the FIT. In order to allow regular handling, the medium will be securely attached to the printed report.

Supervisor:	**Hynek Jiří, Ing.**, DIFS FIT BUT
Beginning of work:	November 1, 2016
Date of delivery:	May 24, 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

[illegible]

Dušan Kolář
Associate Professor and Head of Department

Abstract

This paper describes the steps of designing and development of the Web Application for Information Dashboard Design. The primary challenge is not only to visualize the data on the charts, but also to increase the usability of the final product and to allow programmers to add new widgets quickly and with minimum input required. Visualization tools are being analyzed both from the historical and modern perspectives. Big attention is paid to how human brain perceives information. Charts development tools (in particular, JavaScript libraries) are analyzed and the need in a new application is proven. Development is then described as an iterative process, including application prototype designing and several phases of usability testing and evaluation. As a part of the project, the open-source Vue.js extension for working with charts is developed. Resulting application satisfies all initial requirements and allows to create a new dashboard based on custom design, or using one of the predefined templates.

Abstrakt

Tato práce popisuje analýzu a návrh webové aplikace určené pro tvorbu informačních dashboardů. Nejdůležitějším úkolem je nejen vizualizovat data prostřednictvím grafů, ale také zvýšit použitelnost konečného produktu a umožnit programátorům rychle přidávat nové widgety. Vizualizační nástroje pro prezentaci dat jsou analyzovány jak z historického, tak i z moderního hlediska. Velká pozornost je věnována tomu, jak lidský mozek vnímá informace. Nástroje pro tvorbu grafů (zejména knihovny jazyka JavaScript) jsou analyzovány, z čehož plyne potřeba vyvinout nástroj nový. Součástí diplomové práce je open-source knihovna pro tvorbu grafů založená na knihovnách Vue.js a D3.js. Vývoj samotné aplikace je pak popsán jako iterativní proces, včetně návrhu prototypu a několika fází testování a hodnocení použitelnosti. Výsledná aplikace vyhovuje všem počátečním požadavkům a umožňuje vytvořit nový dashboard založený na vlastním gridovém systémů nebo pomocí jedné z předdefinovaných šablon.

Keywords

Information dashboard, HCI, Web Applications, Data Visualization, Design Principles, User Interfaces

Klíčová slova

Informačny dashboard, HCI, webova aplikace, vizualizace dat, zásady navrhování, uživatelské rozhraní

Reference

PASTUSHENKO, Olena. *Web Application for Information Dashboard Design.* Brno, 2017. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Hynek Jiří.

Web Application for Information Dashboard Design

Declaration

Hereby I declare that this project was prepared as an original author's work under the supervision of Jiří Hynek. The supplementary information was provided by Tomáš Hruška. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

....
Olena Pastushenko
May 23, 2017

Acknowledgements

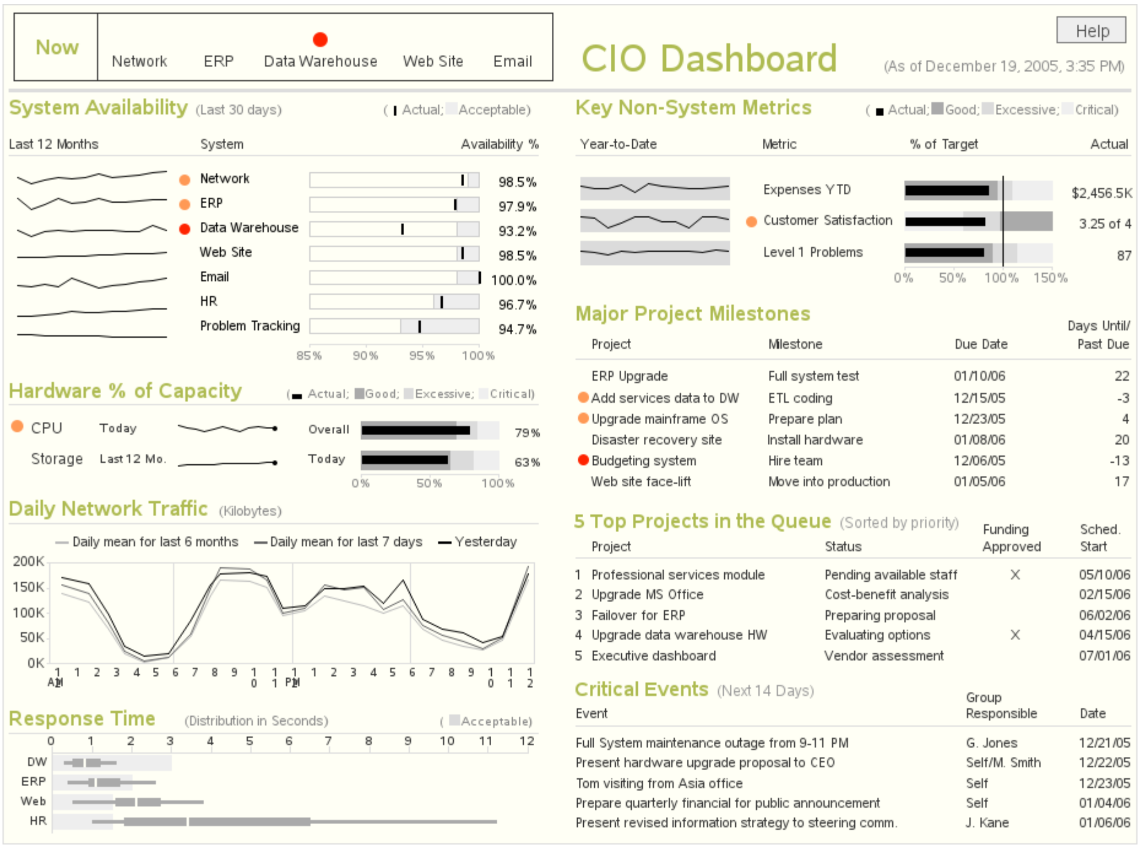
I would like to express my gratitude to my supervisor Jiří Hynek for the useful comments, remarks, borrowing books and engagement through the learning process of this master thesis. And also for always having time for a consultation and discussion. Furthermore, I would like to thank Vero Vanden Abeele, Groep T, KU Leuven for introducing me to the topic of Human-Computer Interaction (HCI) as well for the book's recommendations.

Contents

Chapter 1

Introduction

The focus of this Thesis is to create a Web Application for Information Dashboard Design. Challenging part is to build such an application, which allows creating dashboards quickly and with minimum input needed from a programmer, and at the same time – to make resulting dashboards easily readable by the end user. Therefore, it is important to show data on these dashboards well-arranged and understandable (Figure 1.1).


Now
Network
ERP
Data Warehouse
Web Site
Email
CIO Dashboard
Help
(As of December 19, 2005, 3:35 PM)
System Availability (Last 30 days)
(Actual; Acceptable)
Last 12 Months
Hardware % of Capacity
(Actual; Good; Excessive; Critical)
CPU Today
Storage Last 12 Mo.
Overall 79%
Today 63%
0% 50% 100%
Daily Network Traffic (Kilobytes)
Daily mean for last 6 months
Daily mean for last 7 days
Yesterday
200K
150K
100K
50K
0K
AM
PM
Response Time (Distribution in Seconds)
(Acceptable)
DW
ERP
Web
HR
Key Non-System Metrics
(Actual; Good; Excessive; Critical)
0% 50% 100% 150%

System	Availability %
Network	98.5%
ERP	97.9%
Data Warehouse	93.2%
Web Site	98.5%
Email	100.0%
HR	96.7%
Problem Tracking	94.7%

Year-to-Date	Metric	% of Target	Actual
	Expenses YTD		$2,456.5K
	Customer Satisfaction		3.25 of 4
	Level 1 Problems		87

Major Project Milestones

Project	Milestone	Due Date	Days Until/ Past Due
ERP Upgrade	Full system test	01/10/06	22
Add services data to DW	ETL coding	12/15/05	-3
Upgrade mainframe OS	Prepare plan	12/23/05	4
Disaster recovery site	Install hardware	01/08/06	20
Budgeting system	Hire team	12/06/05	-13
Web site face-lift	Move into production	01/05/06	17

5 Top Projects in the Queue (Sorted by priority)

	Project	Status	Funding Approved	Sched. Start
1	Professional services module	Pending available staff	X	05/10/06
2	Upgrade MS Office	Cost-benefit analysis		02/15/06
3	Failover for ERP	Preparing proposal		06/02/06
4	Upgrade data warehouse HW	Evaluating options	X	04/15/06
5	Executive dashboard	Vendor assessment		07/01/06

Critical Events (Next 14 Days)

Event	Group Responsible	Date
Full System maintenance outage from 9-11 PM	G. Jones	12/21/05
Present hardware upgrade proposal to CEO	Self/M. Smith	12/22/05
Tom visiting from Asia office	Self	12/23/05
Prepare quarterly financial for public announcement	Self	01/04/06
Present revised information strategy to steering comm.	J. Kane	01/06/06

Figure 1.1: Example of a well-arranged dashboard [8]

The first Chapter of the Thesis shows different tools for representing data on a single screen and then Chapter 3 reasons why dashboards are becoming more popular means of data display. They help to summarize data or to monitor activities for achieving a particular goal. There exist a wide selection of tools for creating dashboards, but as shown throughout the paper, none of them satisfy all concepts needed for this particular project.

Chapter 4 shows essential principles of correct dashboards construction. Only when a dashboard is properly designed it can help to perceive all needed information, so it is needed to take into account how the human brain works. In this chapter, it is also shown which widgets are selected for future application as the tools for data representation, and some standard design rules. The primary value is that those graphs are created in accordance with UX (*User Experience*) principles. Those principles are mainly based on the work of Stephen Few. UX defines what impression (experience) users have from using an interface, and how quickly they reach their goals. While analyzing existing software for a dashboard creation it was noticed that mainly they have similar mistakes in designs, and this may have a negative impact on a dashboard usability. That's why there is a need in a way to analyze dashboards usability. It would be possible to export dashboards to XML format, which would contain only parameters needed for the usability analysis and clear grid structure. It is also possible to export created dashboard as PNG image format, to make the analysis from several angles.

Then in Chapter 5, there is an analysis of existing libraries for charts and widgets creation, together with most important principles of web application development. The web application consists of server-side and client-side parts, so it is needed to select appropriate tools for each of them. Since in previous chapters clear need in a new tool for working with graphs was proven, Chapter 6 describes the process of a new JavaScript library creation. The main goal of it is to create a set of reusable Vue.js components based on D3.js and SVG graphs. There are several angles in which practical utility of this library may be proven. First of all, D3 helps to create a huge variety of different data visualization types. But, the question is how usable and understandable they are. Since D3 offers a lot of possibilities, a programmer may be focused more on finding an appropriate coding solution, then on designing an understandable graph. UXgraph solves this problem by providing already predefined and easy to use components. The user needs to specify several parameters only. Vue was selected as a base library because it is fast and provides an opportunity to create new reusable components which can be imported and used as a new custom HTML tag throughout the web page. Components created during this project are gathered into Vue.js library, which is hosted on GitHub as an open-source tool under MIT License. Beta-version is already published as npm (*Node Project Manager*) module, so it is available for all Node.js based projects. Article describing this library was also selected and represented in a for of a poster during the ExcelFit 2017 student's conference. [16]

Chapter 7 contains a description of user and task analysis performed before the application development. As a primary persona, researchers were selected. Since the application is developed directly for their purposes, the primary interface should be adjusted accordingly. So, the user knows the goals of the project, is motivated enough to use the application and looks for a way to generate samples for testing. Application architecture is also described in this chapter.

After the development of the application had been finished, user testing and evaluation were performed. There were two phases of the tests. Formative usability testing was performed after the development of the application prototype. And then at the later stage of the application development, summative usability evaluation was made. It helped to define wrong interface elements at the early stage, when it is easy to make an improvement, and then – to evaluate resulting interface, using selected metrics. The methods, results and improvements made after the testing are described in Chapter 8.

Chapter 2

Analysis of existing tools and methods for representing data on one screen

Data visualization (or representing data on one screen) is a wide concept, which includes some kind of visual transformation of an underlying data [3]. Visualization is a cognitive agent which means that we use it as a way of making sense of complexity. It helps to reduce cognitive overload, even though that is not the main goal. According to Stephen Few [9], the goal is that data should be visualized in a way that leads to understanding. Whatever else it does, it must inform.

2.1 Data visualization history

The concept of using pictures to understand data is an old one and takes the beginning in the oldest discovered rock paintings – Lascaux Cave Paintings (Figure 2.1) which were found in 1940 inside the network of caves in France.

Figure 2.1: Lascaux Cave Paintings.[1]

[1]Source: Wikipedia, Author: Prof saxx

Another ancient example may be Anaximander world map (Figure 2.2), dated approximately 550 B.C. and considered to be the first world map.

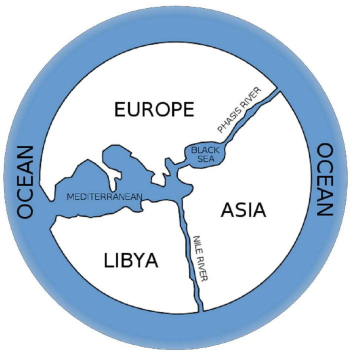


Figure 2.2: Reconstruction of Anaximander's map.[1]

Data visualization principles were widely used by chronologers throughout centuries, even though it took them quite a long period of time to progress from creating tables that contained information to the charts which expressed it graphically [17]. This change was especially noticeable in 16th-17th centuries. A nice example might be allegorical time-maps, which were popular as mnemonics and showed information not chronologically but locating events on a figure corresponding to some characteristics of them. Figure 2.3 shows how Darius of Persia is located under the lungs of the David statue, and these may be understood as a metaphor because during his rule Jews could «breathe freely».

Later on, during 17th and 18th centuries, a lot of graphs which are still widely used, were created. Most of them were originally designed by William Playfair (1759-1823) and Johan Heinrich Lambert (1728-1777) who made the idea of presenting data to a mass audience quite popular. Playfair was the one who invented pie chart and presented the wide use of line charts (examining the import and export differences between Britain and various other countries, Figure 2.4).

Another important landmark is Minard's map of the losses of Napoleon's army in the Russian campaign of 1812 (Figure 2.5). Edward Tufte claimed that it «may be the best statistical graphic ever drawn» [18]. The map showed the size of the army and the path of Napoleon's retreat from Moscow, connecting it together with temperature and time scales.

At the same period a lot of other visualisation methods were invented, such as: first wind-rose graph with the polar coordinates by L. Lalanne, dot map of disease data (cholera) by John Snow and first graphs with variable width. It was also a period of data collection and dissemination in different fields, development of statistical theory and new means of technologies. Lithography started in the 1800s and then resulted in color printing at 1850s, automated recording made possible by James Watt in 1822 and automated calculations by Babbage in 1822, photography and motion developed rapidly as well. [17] All this resulted in a great progress in the field of visual language. People started to use visualization techniques to decide where to build roads and canals, to show changes over time and differences over space. A nice example was created again by Minard in the field of civil engineering. Investigating the reasons for the collapse of the bridge at Bourg-St. Andeol, he used the self-explanatory diagram, which shows the power of visual explanation. It also made possible such purely graphic discovery as Galton's discovery of weather patterns. In

[1]Source: Wikipedia

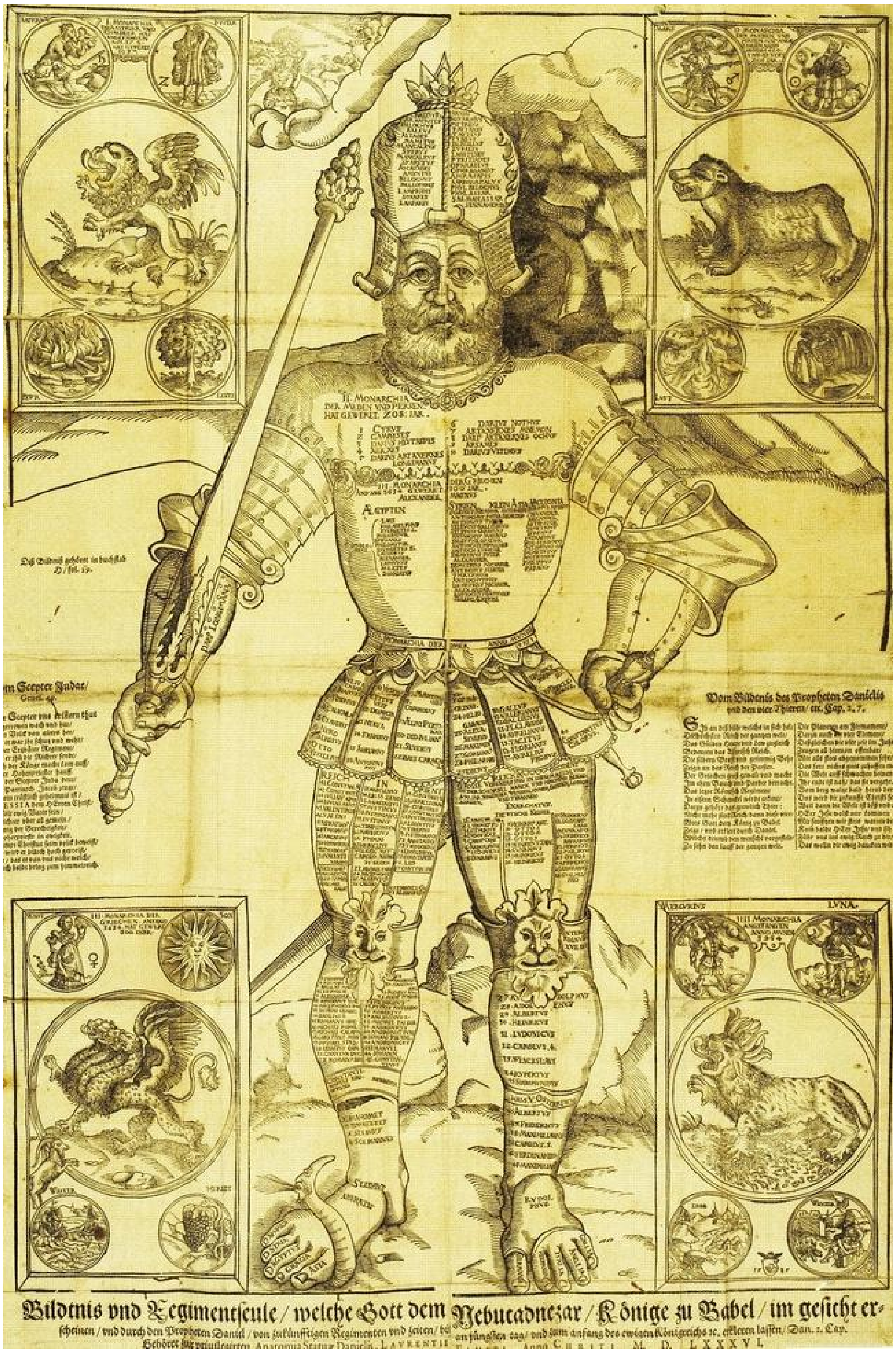

Bildtnis vnd Regimentſeule / welche Gott dem Nebucadnezar / Könige zu Babel / im geſicht erſcheinen

Figure 2.3: Example of allegorical time-map. [17]

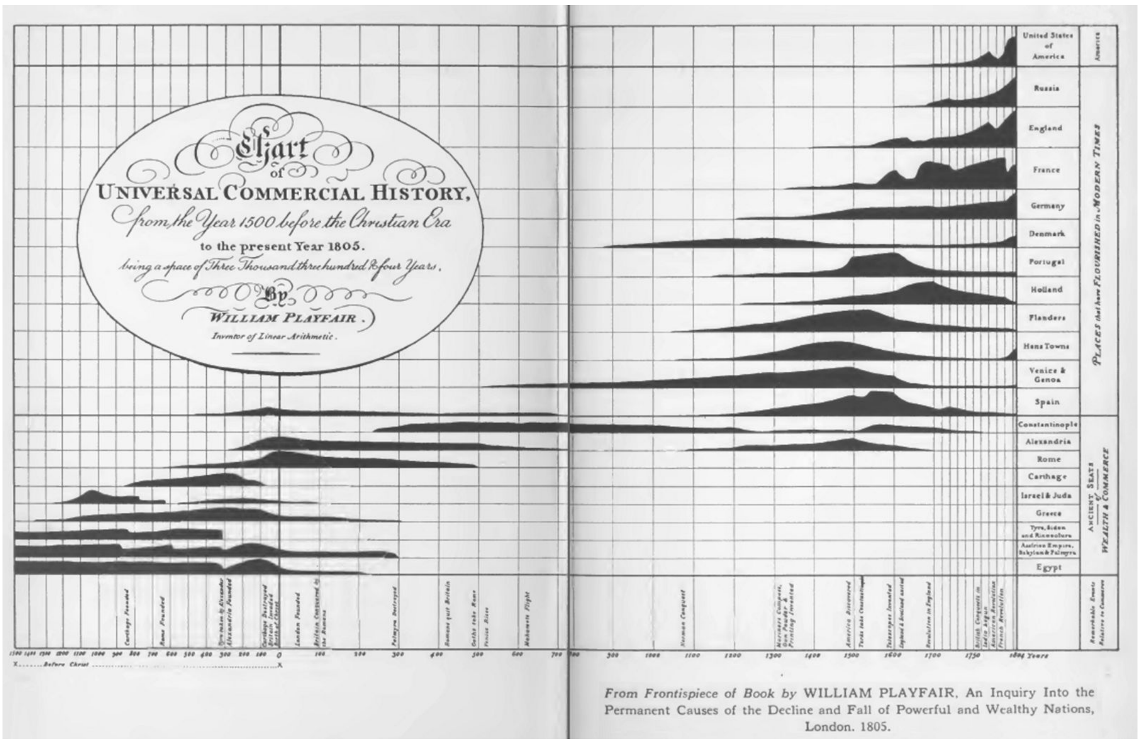


Figure 2.4: The Universal Commercial History, Playfair 1805.
Source: Wikimedia Commons

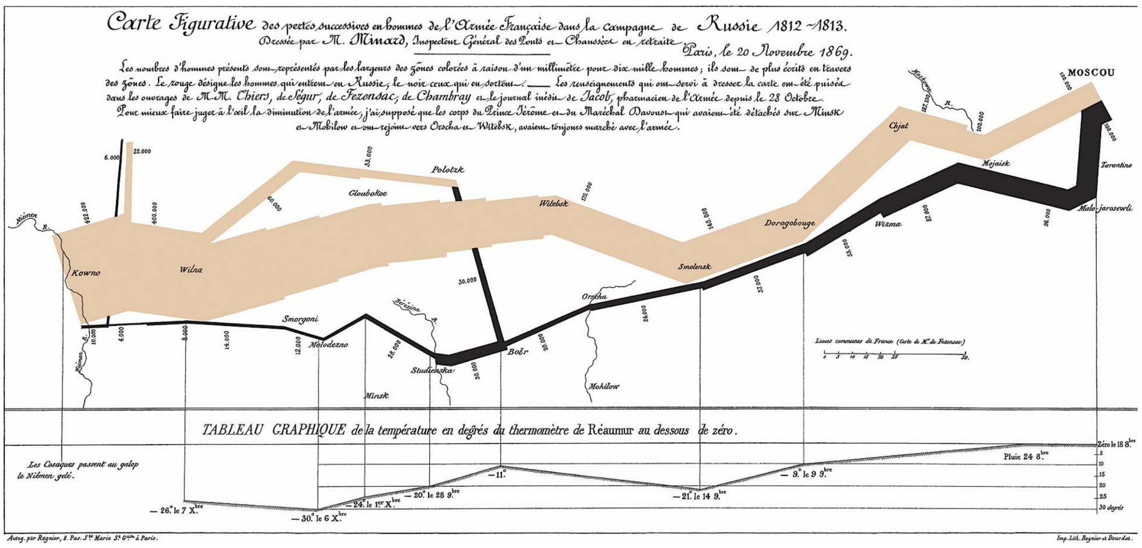


Figure 2.5: Minard's map of the losses of Napoleon's army
in the Russian campaign of 1812. [17]

1861 all weather stations across the Europe (there were 50 of them) were asked to record data (barometric pressure, wind direction, and speed, temperature, rain, cloud) 3 times daily during the December. These data were gathered into 93 maps, containing multivariate glyphs showing all variables. Small graphs with grids and patterns were created from these maps, what made direct comparison much easier (Figure 2.6).

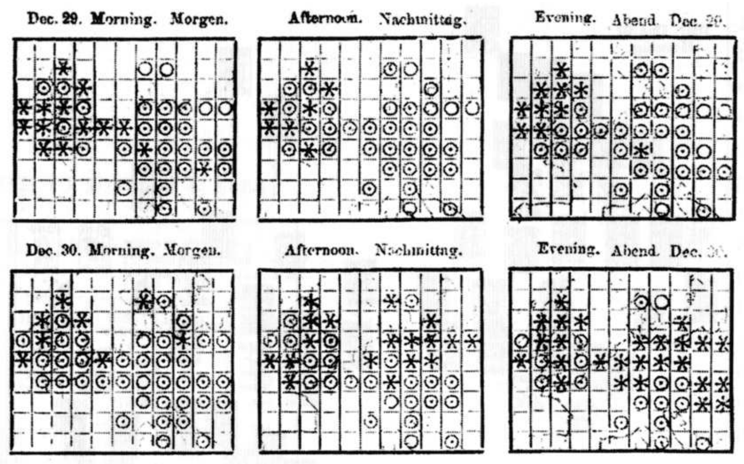


Figure 2.6: Galton's discovery of weather patterns.[1]

Then, with the help of abstractions, large patterns maps were created. These final maps helped to analyze data and to understand how to predict it. Since then visualization techniques increased in popularity [17]. Such visualizations as Distribution of passengers and goods from the Paris railways to the rest of France (1884), State population diagram in the US (1887), Proportions of Catholics and Methodists in the total population (1890) to name just a few, were created.

Nowadays computers made possible to process and analyze larger amounts of data. So these methods are used for representing annual reports, showing flight patterns, describing the most important history events, global income and wealth distribution etc. Today graphs are a vital part of statistical data analysis and a vital part of communication in science and technology, business, education and the mass media [4].

2.2 Types of data visualization

According to Jorge Camões[2], modern data visualization includes following concepts: visual statistics, business visualization, infographics, data art. While speaking about these concepts, it is also important to mention data-scientists who established these terms and wrote the most influenceable papers in the relevant field. All these information is represented on Figure 2.7.

On the very left, there is a group of visual statisticians like Cleveland and Tukey. They do not care that much about the aesthetics of design, but designs they've suggested are still easy to perceive and understand. Cleveland in his paper [4] shows how every graph may be decided into *elementary perceptual tasks*, such as length, angle, area, volume (Figure 2.8) and focuses on understanding how skilled are people each of those tasks [4].

[1]Source: Princeton.edu

[2]Infographics vs. Data Visualization `www.excelcharts.com/blog/infographics-data-visualization/`

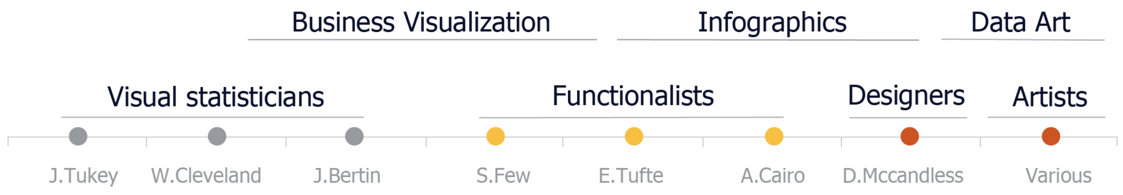


Figure 2.7: Data visualization concepts and data-scientists.

He also made a lot of experiments to prove that bar charts are more effective than pie charts and stacked bars. Paradigm of cleaner, minimalist charts started from his researchers, but still we can see that aesthetic wasn't taken into account.

Then there is a large group of functionalists. They pay a lot of attention to the functional aspects of design and they tend to use them. The first author in that group is Stephen Few. Mainly his concepts of dashboards usability would be covered in the next chapters of this Thesis. His paper was chosen because he is aimed to find a golden mean between functionality and aesthetic in terms of business visualization. He covers a lot of examples of bad visualizations, describing what is the problem and a possible way to avoid it. An example is Figure 2.9, where on the left is shown a typical mistake while creating a graph – adding some unnecessary elements and effects, which results in increasing a time needed to read the information. On the right of the Figure 2.9, the same data is presented but in a way cleaner form. Web application which is a subject of this Thesis is aimed to allow users to create quickly graphs like that, which are easy to perceive.

Also working in the field of functionalism but more concerned on the infographics and the art side of data is Alberto Cairo. He claims that effective infographics can still be fun «if their designers don't forget that their fundamental goal is to make the public better informed» [2]. As one of the examples, he uses Chinese worldwide exports map (Figure 2.10). Due to his book, great visualization should be truthful, functional, beautiful, insightful and enlightening.

There is also a group of designers, where function becomes decoration. Here we can meet the possible sacrifice of effectiveness for a pleasing design. This approach still cannot be called «art» though. The main representative of this group is David Mccandless.

And then, the last group displayed in the image consists of pure artists. They are using data as just another source for their art creations. A nice example may be composite photos of airplanes in flight by Mike Kelley Figure 2.11.

Another example is a project which was getting bees to construct geographic maps. Here was used a principle that a colony will follow the queen bee and build a hive based on the pheromones that she releases, so if you move the queen the others in the colony act accordingly (Figure 2.11).

Summing up all above, it might be said that every visualization developer is responsible for finding a balance between the requirements of utility, soundness, and attractiveness within given constraints [15]. It was first written in the book De Architectura in 25BC by Roman architect Vitruvius. Utility here means usability and functionality and can be measured by efficiency. Soundness represents the quality of the visualization presentation algorithm. And finally, attractiveness corresponds with aesthetics.

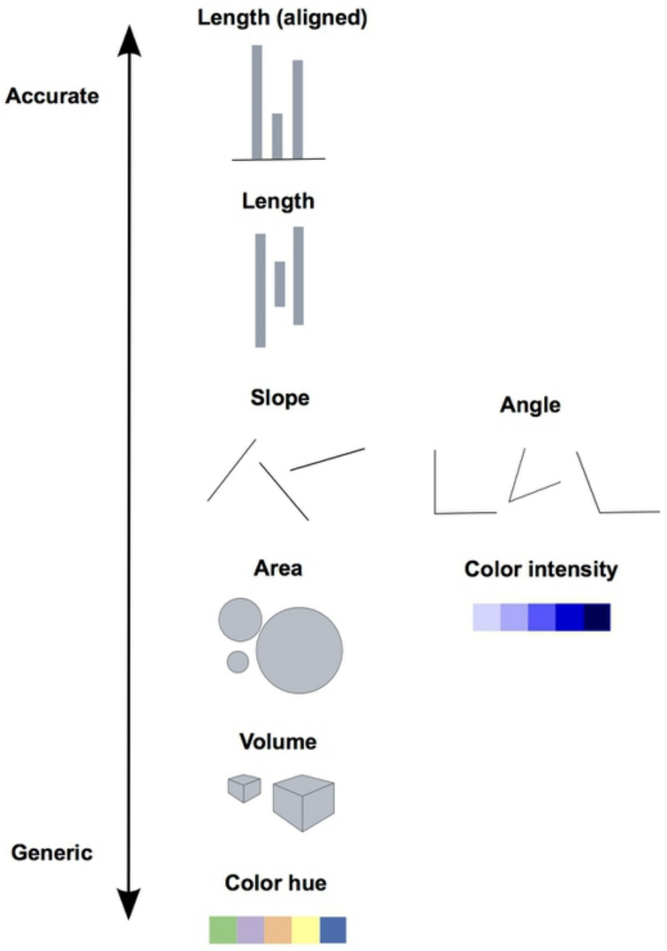


Figure 2.8: Elementary perceptual tasks by Cleveland.
Source: Introduction to Data Visualization

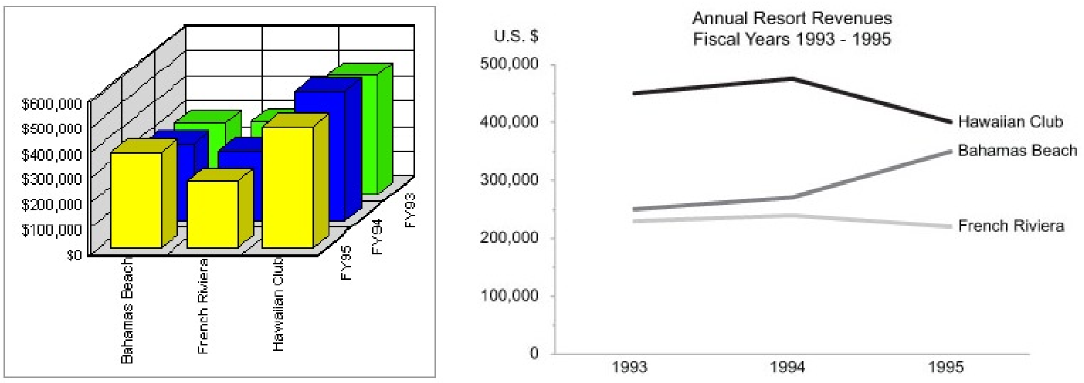


Figure 2.9: Bad example of 3-D bar graphs (left) and improved version (right) according to Few [8].

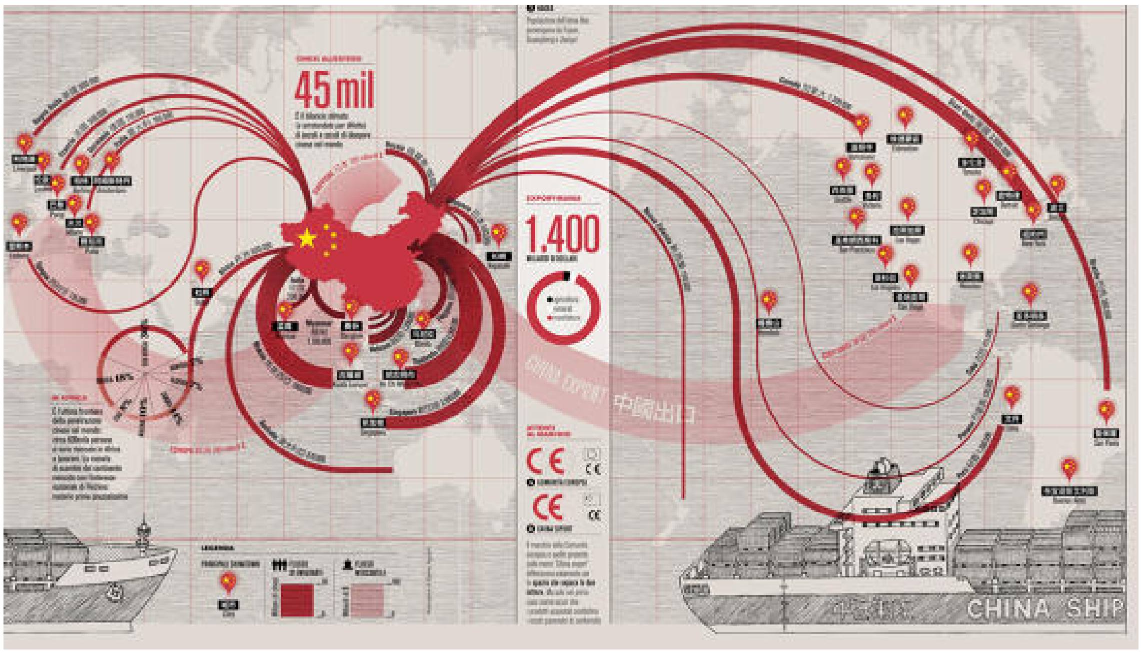
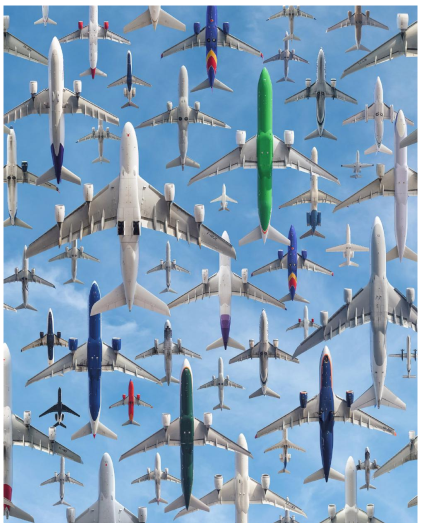


Figure 2.10: Chinese worldwide exports map as an example of effective infographics.[2]



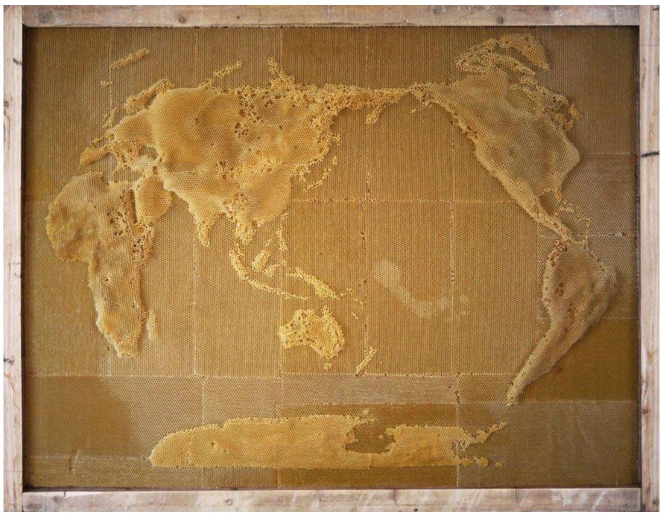
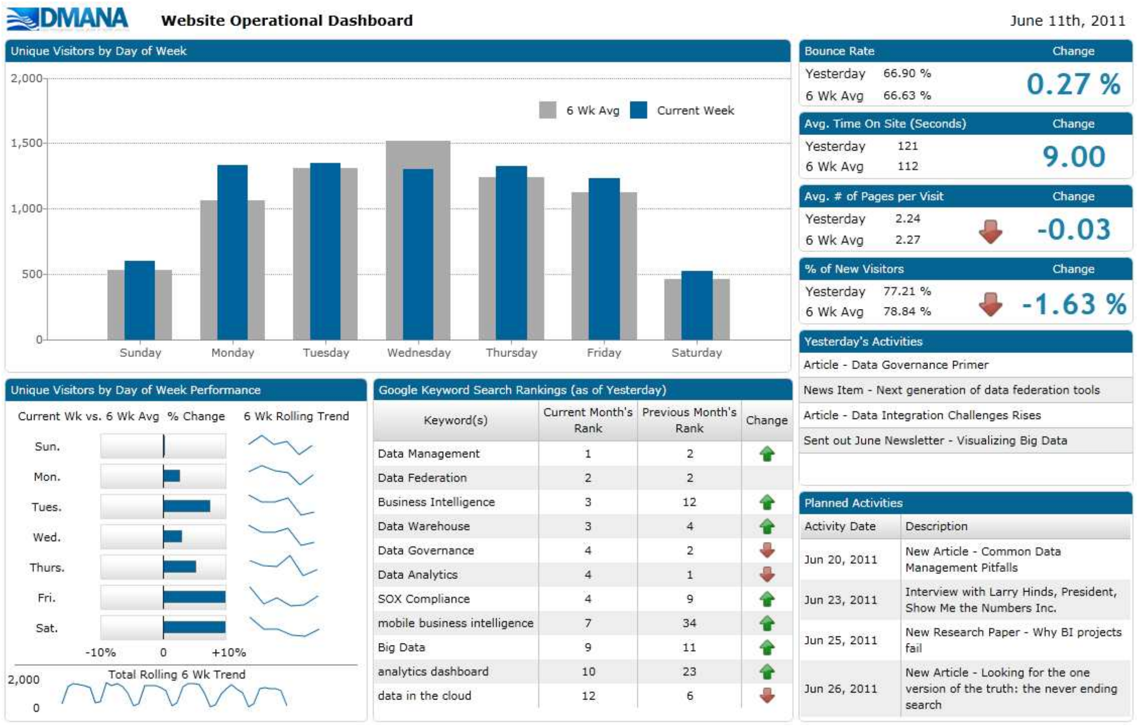


Figure 2.11: Examples of data-art.
Source:
http://flowingdata.com/category/visualization/artistic-visualization/

Chapter 3

Dashboards

There exist several definition of a dashboard. Different papers about dashboards, as well as websites of dashboards' software providers, agree that a dashboard must include graphical display mechanisms. But the most suitable definition is made by Stephen Few in his article for Intelligent Enterprise magazine: «A dashboard is a visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance» [7]. A dashboard is not some particular technology used for specific type of information. It is a type of display, a form of a presentation designed to communicate (Figure 3.1). Dahboards are the part of a bigger concept, called *Business intelligence*.



Bounce Rate		Change
Yesterday	66.90 %	0.27 %
6 Wk Avg	66.63 %	

Avg. Time On Site (Seconds)		Change
Yesterday	121	9.00
6 Wk Avg	112	

Avg. # of Pages per Visit		Change
Yesterday	2.24	▼ -0.03
6 Wk Avg	2.27	

% of New Visitors		Change
Yesterday	77.21 %	▼ -1.63 %
6 Wk Avg	78.84 %	

Google Keyword Search Rankings (as of Yesterday)

Keyword(s)	Current Month's Rank	Previous Month's Rank	Change
Data Management	1	2	▲
Data Federation	2	2	
Business Intelligence	3	12	▲
Data Warehouse	3	4	▲
Data Governance	4	2	▼
Data Analytics	4	1	▼
SOX Compliance	4	9	▲
mobile business intelligence	7	34	▲
Big Data	9	11	▲
analytics dashboard	10	23	▲
data in the cloud	12	6	▼

Yesterday's Activities

- Article - Data Governance Primer
- News Item - Next generation of data federation tools
- Article - Data Integration Challenges Rises
- Sent out June Newsletter - Visualizing Big Data

Planned Activities

Activity Date	Description
Jun 20, 2011	New Article - Common Data Management Pitfalls
Jun 23, 2011	Interview with Larry Hinds, President, Show Me the Numbers Inc.
Jun 25, 2011	New Research Paper - Why BI projects fail
Jun 26, 2011	New Article - Looking for the one version of the truth: the never ending search

Figure 3.1: Example of a dashboard.[1]

[1]Source: Dashboard Insight

3.1 Business intelligence

Business intelligence (BI) is the part of information technology that focuses on reporting and analysis. It is a very important tool to use in order to reduce cost and time. It may be also defined as an interactive process for exploring and analyzing structured, domain specific information to discern business trends or patterns, thereby deriving insights and drawing conclusions. The business intelligence process includes communicating findings and effective change. Domains include customers, suppliers, products, services and competitors [10]. There are typically four types of presentation media: dashboards, visual analysis tools, scorecards, and reports. They all serve a similar purpose, helping people to find trends, correlations, and patterns. But in the same time, all of them differ with some unique attributes.

Back in 1980's, there were developed several executive information systems (EIS). Their main purpose was to display financial measures on simple graphical user interfaces. But in those times they didn't become popular, as a technical side was quite difficult and required a lot of information from different sources, which quite often appeared to be incomplete [8] Later on, during 1990s data warehousing, BI and online analytical processing became more popular. The main emphasis was now put on the technology itself and not on the methods of usability and availability improvement. Primary persona was considered to be not an average executive, but proficient in computer technologies user. And we might say that the right data in the wrong form say little of importance.

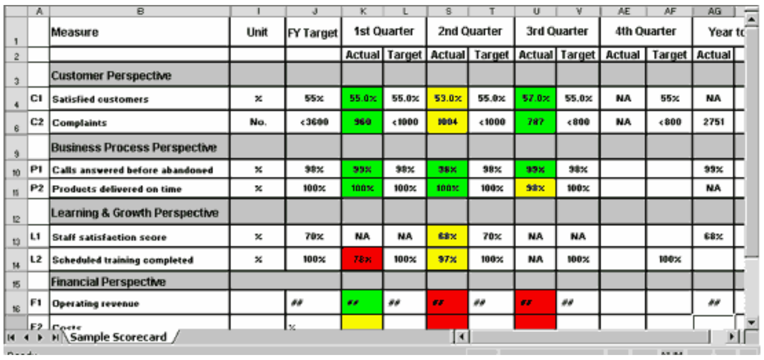
The situation changed in 2001 after the fall of Enron. Enron was a US based corporation, famous as innovator and technology powerhouse. It gave an illusion that it was a steady company, but after an investigation, it appeared not to be so. This was made possible by great frauds with accounting and surfacing information about hiding losses [11]. The aftermath of this case showed the importance to monitor all inner processes and of being able to present such information to shareholders. As a result, different BI providers started to offer dashboard services. The increasing popularity of dashboards was documented in a research done by AMR Research inc: «more than half of the 135 companies ... recently surveyed are implementing dashboards». But even taking into account rising popularity of dashboards it was still difficult to give a precise definition of what a dashboard actually is.

Stephen Few also defines next key characteristics of a dashboard:

- All the visualization fit on a single computer screen. This is important to ensure that all information may be perceived at one glance.
- It shows the most important performance indicators, important in achieving certain goals.
- Interactivity such as filtering may be used in a dashboard, but shouldn't be required.
- They are designed for wide circle of users and are easy to use.
- Updates are automatically.

Dashboards are composed of different data visualization tools, such as maps, graphs, charts, grids, and gauges. So, taking into an account all said above, a dashboard may be called an information management tool that visually tracks, analyses and displays *key performance indicators* (KPI). They evaluate the state of the organization or business in a particular activity. Dashboards are highly customizable and may fit the needs of different companies and departments. They always show real-time information, tracking and combining information from different sources. To do so, a dashboard is usually connected to different files, API's and services, even though for the end user it shows all information on the one screen in a form of tables and diagrams. To make the difference between dashboards and other types of business visualization clearer, key features of the others are covered next. Visual analysis tools represent data on the one image as well, but they allow to pick different ranges, products, detalisation levels. The main difference here may be that these tools are always highly interactive and may contain lots of scrollbars or zoom inside its sections. Since usually not real-time data is displayed there but mainly some time-lapses, they can be used to define key performance indicators for future usage in the dashboards. A good example of such a tool might be research on homicides in the D.C. by Washington Post, held between 2000 and 2011, which was aimed to find out what exactly happened in each case.

Then scorecard (Figure 3.2) is a tabular visualization of measures and their respective target with visual indicators to see how each measure is performing against their targets at a glance. They are not interactive and so have only one depth level, containing a measure, its target and a visual indication of a status. Usually, it may be used as a part of a dashboard or visual analysis tool.



	A	B	I	J	K	L	S	T	U	V	AE	AF	AG
1		Measure	Unit	FY Target	1st Quarter		2nd Quarter		3rd Quarter		4th Quarter		Year tc
2					Actual	Target	Actual	Target	Actual	Target	Actual	Target	Actual
3		Customer Perspective											
4	C1	Satisfied customers	%	55%	🟩 55.0%	55.0%	🟨 53.0%	55.0%	🟩 57.0%	55.0%	NA	55%	NA
6	C2	Complaints	No.	<3600	🟩 960	<1000	🟨 1004	<1000	🟩 787	<800	NA	<800	2751
9		Business Process Perspective											
10	P1	Calls answered before abandoned	%	98%	🟩 99%	98%	🟩 98%	98%	🟩 99%	98%			99%
11	P2	Products delivered on time	%	100%	🟩 100%	100%	🟩 100%	100%	🟨 98%	100%			NA
12		Learning & Growth Perspective											
13	L1	Staff satisfaction score	%	70%	NA	NA	🟨 68%	70%	NA	NA			68%
14	L2	Scheduled training completed	%	100%	🟥 78%	100%	🟨 97%	100%	NA	100%		100%	
15		Financial Perspective											
16	F1	Operating revenue		##	🟩 ##	##	🟥 ##	##	🟥 ##	##			##
	F2	Costs		%	🟨		🟥		🟥				

Sample Scorecard

Figure 3.2: Example of a scorecard.[8]

Reports usually do not content any graphics but still may use visualization to highlight needed data. Mainly it presents data, containing from text and numbers, in a table, and is optimized for printing.

3.2 Dashboards taxonomies with examples

Categorizing the dashboard plays an important role in its development, as it helps to distinguish and select main features and concepts of the design. Of course, there are a lot of common elements for all types, but differences in the details may play a significant role in the perception of the information by a target user group.

According to Stephen Few, categorizing dashboards by the role they play relates mostly to a dashboard's visual design [8]. So the most common division is strategic, analytical, operational or mixed dashboards. Within this taxonomy, choosing the type of a dashboard depends on the type of the problem which needs to be solved.

1. *Strategical dashboard* may be called any dashboard used for making a strategic decision. They provide a quick overview of all processes and are aimed to monitor the

opportunities for improvement and the general health of the business. Since strategic managers don't usually provide any type of deep analysis, dashboards of this type mainly do not provide any type of interactivity, on the contrary – the most simple mechanisms work best for them. Too many details can also play a bad role here, since they may distract a decision maker from his direct goals. They also do not need any real time data, since decisions in a business are not being made on a go. But monthly, weekly or daily snapshots might be really in use.

2. *Analytical dashboards.* It usually follows tactical, short and medium-term objectives. Data here demands a lot of contexts and extensive history. This types of dashboards mainly do not need to present real-time data, but more monthly or weekly static snapshots. However, it is quite important to give the user a possibility to explore some data or time period in more details, when needed. So, interactivity (such as drilling down into the details, examining some cases) is highly welcomed. It is important not just to show some facts about the business (for example that the sales are decreasing), but also to provide the end user with all needed means to find out the reason of this process.

3. *Operational dashboards* show the data used by business managers or the general workforce. The main difference from the other types is that data here needs to be dynamic, so mostly presented in real time or near real time. They are used to monitor operations and activities so that end user may pay attention to some certain changes. For example, a robotic arm which makes some fixes is a lack of important details, this needs to be reacted immediately, and not on the next day, when a daily time-lapse would be available. It also helps to maintain processes within certain borders. So that when any of the parameters goes beyond the predefined border the relevant user gets notified immediately. Since some situations may be the cases of the emergency, data presentation should be simple and clear. Though at the same time it is extremely important to have all needed details. For example, if a delivery of a hight priority is about to miss a deadline, some high order statistics is not enough, it is needed to show all details about the particular order and the reasons for the delay. So, it is acceptable to have both interactive and static displays for operational dashboards. On a static one, all needed information should pop up and grab an attention in the case of emergency, while interactive dashboards should provide an end user with all needed means to investigate the case quickly by himself.

Another possible classification is by data domain. It relies on the source of data which is going to be displayed, or it may also be said that it is specific for the business departure or field where it might be used. The main types within this taxonomy are:

- *Executive dashboards* (Figure 3.3). Used mainly by CEOs and top executives in order to monitor KPI, different metrics and other data on the most higher, general level. Below is the example of KPI dashboard, found on the website of one of the dashboards' software providers:

- *Healthcare dashboards.* Usage of this kind of a dashboards may not be that obvious in this industry, but in fact, they are widely used to display large amounts of data, showing general status of the healthcare delivery. As a result client's satisfactory may be understood, together with ways of service improvement. An example of measures may be the average length of stay and lab turnaround time.

- *Marketing dashboards* (Figure 3.3). They may be used by social media specialists, web analytics, content marketers. They can help to evaluate marketing campaigns, measuring ROI and. other metrics.
- *Sales dashboards*.
- *Manufacturing dashboards*.
- *Human resources dashboards*.

There may also be other taxonomies mentioned. Such as:

- **type of data**: quantitive, non-quantitive,
- **types of measures**: balanced scorecard, six sigma, non-performance,
- **span of data**: enterprise-wide, department, individual,
- **update frequency**: monthly, weekly, daily, hourly, realtime or near realtime,
- **interactivity**: static display, interactive display,
- **mechanisms of display**: primary graphical, primary text, their combination.

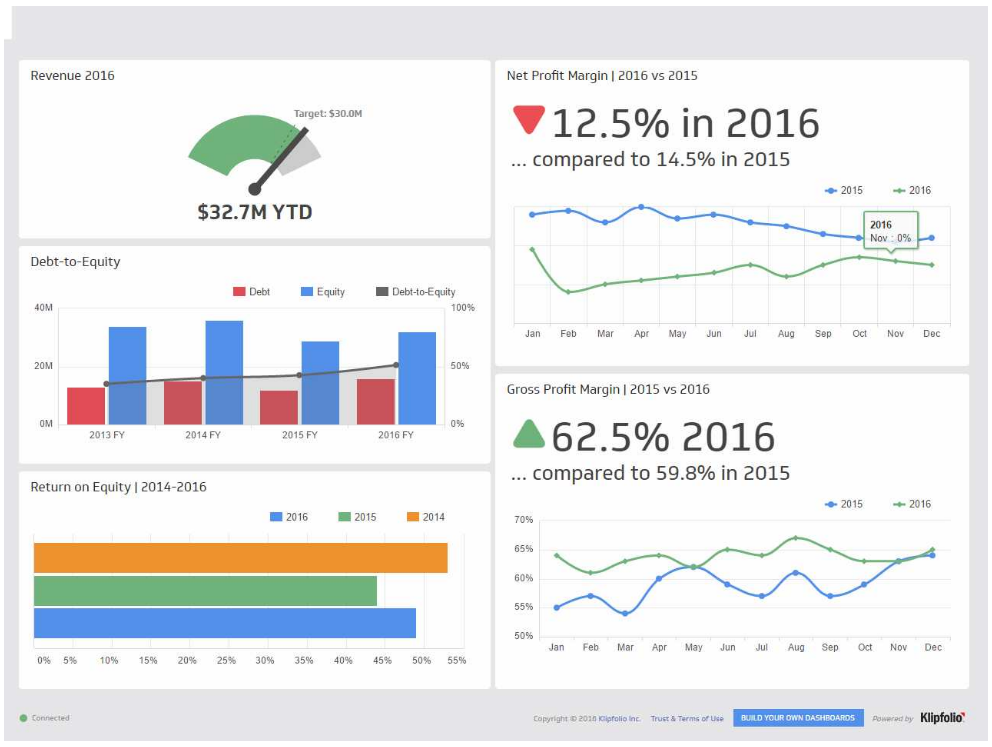
3.3 Motivation for creating dashboards

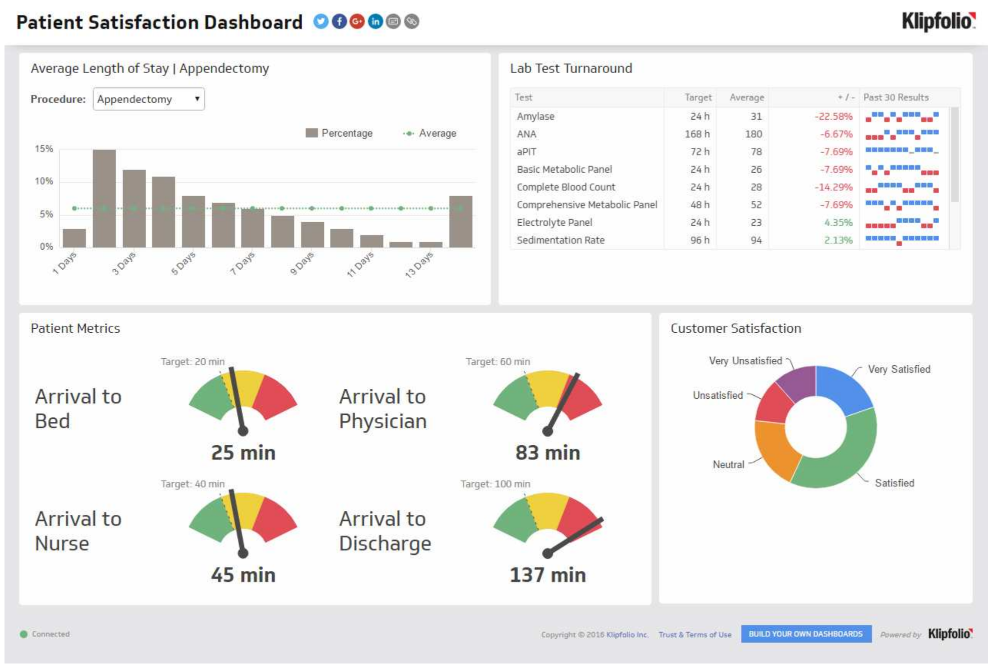
Motivation refers to psychological processes that are responsible for initiating and continuing goal-directed behaviors. People have been monitoring businesses without dashboards for ages. As a result, decision makers in companies often suffer from too many data which have too little information which is delivered too late to make an effective decision. These problems may be solved by using dashboards, developed in accordance with the best practices. Then they are providing a powerful mean to present information. Some of the most common reasons for using a dashboard are:

- better visibility, as it allows to monitor whole business, process or an operation just at one glance,
- time saving, because when it is needed to get some data, there is no need to wait for a report – everything needed may be received from a dashboard.

Making more deep research about the motivation, all other reasons may be divided into four groups: advantages for the users, a motivation of workers, rich data usage, improvement of communication and better business planning [6].

- **Advantages for the user**. The main thing is that dashboards make possible to monitor a status of several areas on one screen, allow to represent this status graphically rather than in textual form. An important feature is that alerts about some critical conditions may be easy-noticeable, and moreover – may allow a possibility to click to see more details. And all this is done in such an intuitive form, that no training is required. Dashboards allow users to get all needed information without signing in some web service, printing and looking through reports or getting through emails.





Test	Target	Average	+ / -	Past 30 Results
Amylase	24 h	31	-22.58%	
ANA	168 h	180	-6.67%	
aPIT	72 h	78	-7.69%	
Basic Metabolic Panel	24 h	26	-7.69%	
Complete Blood Count	24 h	28	-14.29%	
Comprehensive Metabolic Panel	48 h	52	-7.69%	
Electrolyte Panel	24 h	23	4.35%	
Sedimentation Rate	96 h	94	2.13%	

Figure 3.3: Examples of an executive (on top) and marketing (bottom) dashboards. Source: Klipfolio dashboards examples

- **Motivation of workers**. The well designed dashboard can help focus on what is really important, providing different interfaces and functionality based on position and circle of interests. It may help to quickly show employees that their contribution counts in term of business growth, and also to motivate them to use goals, competition, incentives and other methods of gamification.

- **Rich data**. This is achieved by blending data from different sources and allowing both historical and real / time data, as well as both detailed and aggregated data. Effective dashboard design utilizes colors, symbols and visualization techniques, such as tables, line charts, and gauges) to highlight important data points.

- **Improvement of communication**. Using a dashboard allows better coordination between departments, and between managers and employees. A key feature is that you don't need to be a marketer or analyst to understand what is represented on a dashboard. It helps to keep every member of a business on the same page and allows sharing. There are several ways of sharing a dashboard, depending on the privacy level of used data. For example, a link for a public sharing may be created, when a company wants to show its success to potential customers, though dashboard with internal budget calculations may be shared only with several high-level managers. Dashboards are getting more popular because they allow to create a virtual work environment and as a result – make it easier for teams to collaborate. For example, analysts may work in the background, providing more data to show on the dashboard, while users can access and work with generated data anytime and anywhere.

- **Business planning**. The dashboard is also a very convenient tool for communicating a strategy. It helps to ensure that everyone uses the same data, metrics and works towards the same strategy. When storing dashboard on a cloud, it can be accessed anytime and from any device (wallboard, computer, tablet or a phone). Dashboards make reporting more efficient, as they may generate them automatically at any time (and not only in the end of the month).

3.4 Review of the existing software for dashboard creation

Since nowadays dashboards are getting more and more popular, the number of software providers for their creation is also increasing rapidly. Before looking into more details about most popular ones, it is important to categorize them. The main difference is in where the data is stored and from what kind of devices it may be accessed. The software may be installed directly on a computer or internal network, on a cloud service or stored within a native mobile application. When storing in a cloud, data may be accessed from any device with the connection to the internet.

While making an analysis of a software for dashboard creation for this project, several test accounts were created to see the demo versions of existing tools. None of them satisfied all needed items which are going to be needed to create testing data quickly. Some application were missing test data. Some of them needed additional installation, and this project is going to be about a web application so that no installation is needed and it is available for any device and OS. Also, there were some design mistakes (according to rules described by Few) [8]. And the most important thing is that existing software do not allow to create a custom export format. All above shows the need for creating a new web application, even though there is quite a big selection of similar ones.

Chapter 4

Principles of dashboards design

To increase dashboards usability for an end user, it is important to understand it's purpose, how it will be consumed. For example, there might be different dashboards for a technician and executive because of their different technical background, acronyms they are used to and information they need. And different purposes require different visualizations. According to Noam Iliinsky [13], to create a successful dashboard, one must be able to articulate the purpose and focus firstly. Then, to select minimum info of a correct information needed to present on a dasboard, and make a clear structure. And then, as a last step of the development process, it is needed to make a formatting useful. Some rules of dashboards construction are different for different types, but there are some general and fundamental rules which should be taken into account in any case. Such rules are the subject of this chapter of the Thesis.

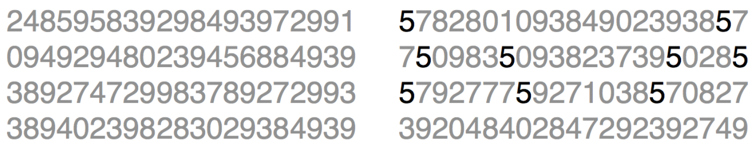
4.1 Ergonomics of the iteration

The big mistake of any software development process is to focus on the features development only, not carrying in mind the general application architecture and vision. To build complete and of high usability dashboard, it is important to take into account also how people perceive and think. At the same time, it is important not to end up developing a business software which is more entertaining then useful.

Science that deals with designing things so that they fit the people who use them is called *ergonomics*. Main principles are described in book by Jeff Johnson [14]. Only those of them which are relevant to dashboards and are going to be used for this project. Main ergonomics principles which are relevant to them are going to be used for the project and are listed in this chapter.

People have attentive and preattentive ways of perceiving the information. Preattentive processing is a subconscious accumulation of information from the environment. Important information should be easily distinguished from the irrelevant parts, to help user to accumulate it quickly. An example is shown in Figure 4.1. When given a task to calculate the amount of „5“ within certain amount of other numbers, it is much easier to do when target number is emphasized.

More of these principles are combined in so-called *gestalt psychology* [14]. One of the main principles says that in perception the whole is different from the sum of its parts. Applying to dashboards, this means that it is not enough to pay attention only to the design of the graphs. Whole dashboard should be developed in a way which helps to keep



```
24859583929849397299 1    57828010938490239385 7
09492948023945688493 9    75098350938237395028 5
38927472998378927299 3    57927775927103857082 7
38940239828302938493 9    39204840284729239274 9
```

Figure 4.1: Explanation of preattentive processing.

consistency for all widgets, and clear grid structure. Other gestalt principles explain rules of clustering perceived items into groups, depending on their similarity, connection, enclosure or symmetry. These principles may be used to design correct graphs or to choose a suitable type of graph for a particular information. Nice example of applying gestalt principles during designing a graph is shown on Figure 4.2.

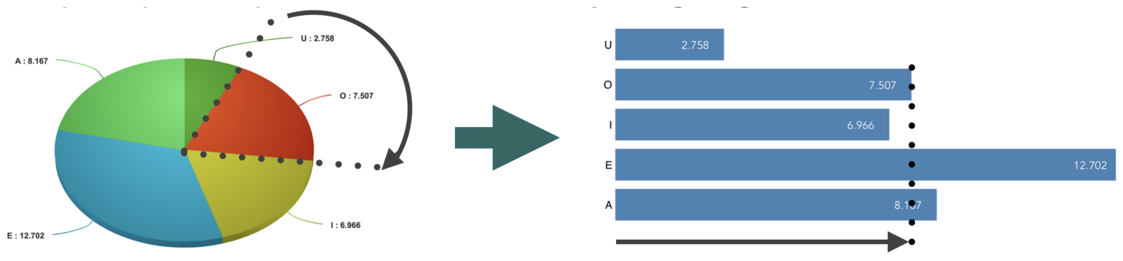


Figure 4.2: Pie charts are used quite commonly, but they are less effective than the bar charts. Whereas a bar chart uses the line length to encode the quantitive information, pie charts use two-dimensional areas of the slices and their angles, and our visual perception experience troubles comparing angles and 2-D areas.

Besides pre-attentive attributes, there are other factors which may influence user's interaction with a software, such as their experience, goals, and context. For example, most common colors for the status icons are green (notification), yellow (warning) and red (error). But for color blinded people two of them might look almost similar. That's why it is better to use several hues of the same color, then different colors, to distinguish statuses. Another use case is a difference in a context: percieved information would be different for a user who have a lot of time and quiet surrounding during the interaction with a dashboard, from the one who needs to get all needed information on a go.

4.2 Dashboard display media chosen for the application

To reach the full potential of a dashboard and clear communication with a user, it is needed to use appropriate and well-designed display medium. Type of medium depends on the nature of the information, its message and the need of the end users. Selection of the display medium for this thesis was made based on the following principles:

- It must be able to efficiently present information when it is displayed on a small screen, without additional scrolling needed.
- It must be the best means to display the most commonly used in dashboards information.
- It must allow to create new dashboards for testing purposes with minimum efforts.

After analyzing different display media, the future library for the application was divided into the following categories:

- graphs,
- icons,
- text.

Following sections contain the detailed description of each category.

4.2.1 Graphs

Graphs are the most commonly used dashboard display media [8]. According to the main design principles which are described in the next parts of the Thesis, the best way to provide information for the most types is to display quantitive data in the form of a 2-D graph with X and Y axes. For this Thesis six types of graphs were selected, based on their usability. They were preferred because they can provide information more effectively than the other alternatives, and at the same time – are quite simple comparing to the others. This is an important feature both while development and future use for the quick test data creation. For example, the pie charts are used quite commonly in the dashboards, but they are less effective than the bar charts. Whereas a bar chart uses the line length to encode the quantitive information, pie charts use two-dimensional areas of the slices and their angles. As it is shown in the Gestalt psychology principles, described in the previous chapter, our visual perception experience troubles comparing angles and 2-D areas. There is a bigger group of graphs which use 2-D space to encode quantitive values, called *area graphs*. None of them where selected for this Thesis. Following list shows types of graphs which are going to be used in the application.

- **Bullet graphs** (Figure 4.3). This type of graphs was developed by Stephen Few specifically for dashboards [8]. Their main purpose is to replace gauges and meters with a type of a chart which consumes less space in a dashboard and is more intuitive at the same time.

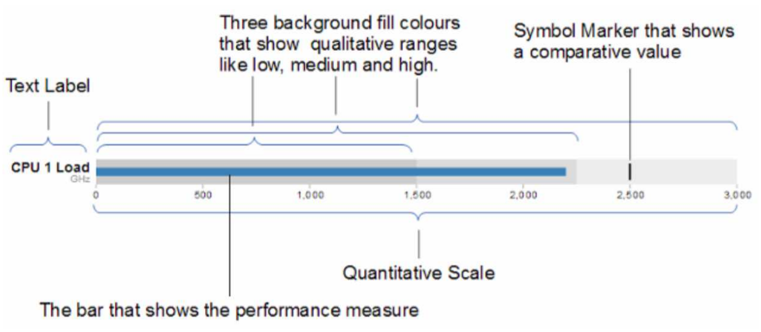

Three background fill colours that show qualitative ranges like low, medium and high.
Symbol Marker that shows a comparative value
Text Label
CPU 1 Load
GHz
0
500
1,000
1,500
2,000
2,500
3,000
Quantitative Scale
The bar that shows the performance measure

Figure 4.3: Bullet graph example.[8]

- **Bar graphs (horizontal and vertical)**. These graphs are also used to display multiple instances of one or more key measures. Bar graphs are most effective when showing measures associated with items in one category. It is also important to use them only for appropriate data. There are nominal (North, East, South, West), ordinal (1st, 2nd), and interval (like days or months) scales. And bar charts should be used for the last ones. Following is the example of a typical bar chart.

- **Stacked bar graphs (horizontal and vertical)** (Figure 4.4). This is a variation of a bar chart, which is better readable when you need to show several instances of a whole, but with the emphasis on a whole. Next image shows an example of such usage.

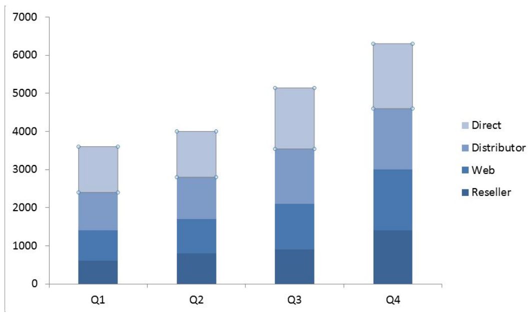


Figure 4.4: Stacked bar graph example.

- **Combination bar and line graphs** (Figure 4.5). This one is recommended to use when some data should be displayed with the emphasis on a comparison of individual values, and some – with an emphasis on the overall shape of the data.

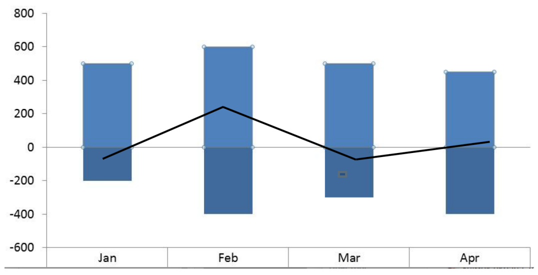


Figure 4.5: Combination bar and line graph example.

- **Line graphs** are the best ones in showing the shape of data movement and the dynamic. Since everything else is eliminated, it helps to create visual patterns which are easy for understanding.

- **Sparklines** (Figure 4.6). This type of graph was invented by Edward R. Tufte and defined as «data-intense, design-simple word-size graphics» [19]. They are a good fit for dashboards, as they provide condensed forms of data display, helping to show some measure within the context.

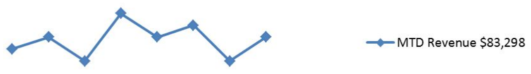


Figure 4.6: Sparkline example.

4.2.2 Icons

Icons are simple images that communicate clear and simple meaning [8]. They are not always needed in a dashboard, but they are an important element to emphasize something (alert icons), show status (on/off icons) or dynamic (up/down icons). The main challenge with icons is to make them simple but noticeable at the same time. Ammount of alert levels should be limited to two, or even better – to one. The reason beyond this is that a single alert icon catches the eye a way more effectively than multiple alerts, because it is easier to notice, and moreover no time is spent on comparing the colors of the multiple icons. Also, it is important to select effective colors based on the information in Chapter 4. Since 10% of males and 1% of females are color-blind, a solution that works for everyone is using different saturations of the same hue, rather than different hues for the alert icons. Different hues (such as green/red, for example) may be used for status or dynamic icons, though when symbol itself is conventional enough.

4.2.3 Text

This element is included into all dashboards, even if they are fully graphically oriented. Some information is simply more readable textually. Other type of information needs text for explanation (such as categorical labels for the items on graphs). Fully textual display medium is appropriate when it is needed to show a single measure alone, without comparing it to anything. When there is a need to arrange a set of information in a particular manner to communicate clearly, tables are considered to be the best display medium [8]. Tables arrange data into columns and rows, which can help when being designed properly, to analyze data easier and to perceive information quicker.

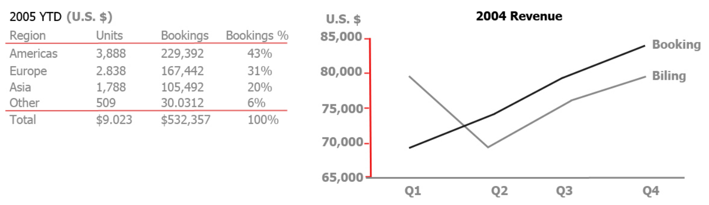
4.3 Best user experience practices applied to dashboards

In previous chapters rules for separate elements were described. This chapter is about general rules and design principles, which should be applied to a dashboard as a whole. Some of them are used as quantitive measures for the framework of dashboards evaluation, which is described in Chapter 4. In a paper by Edward R. Tufte the *data-ink ratio* concept was introduced [18]. Everything that is displayed may be divided into data and visual content which is not data (*non-data*). Due to this concept, when quantitive data is printed, *data-information* is when the ink is changing as the data change. It is the non-erasable core of the graphic, while usage of the non-data ink should be minimized. So, the data-ink ratio is equal to data-ink divided by total ink used to print the graphic. And this ration should be aimed to be equal 1. Since this Thesis is about software, *ink* may be replaced by *pixels*, but the formula would be the same:

$$data\,pixels\,ratio = \frac{total\,pixels\,used\,for\,the\,graph}{data\,pixels} \tag{4.1}$$

Figure 4.7 shows the difference between data-information and non-data:

The first step to improve this ratio is reducing the non-data pixels, and the next one is enhancing the data ones. All unnecessary non-data pixels on a dashboard should be eliminated, and the ones which are needed to make the information readable should be de-emphasized by making them visible enough just to do their job, but not catching the eye. On the contrary, the most important data-pixels should be highlighted. Figure 4.8



2005 YTD (U.S. $)

Region	Units	Bookings	Bookings %
Americas	3,888	229,392	43%
Europe	2.838	167,442	31%
Asia	1,788	105,492	20%
Other	509	30.0312	6%
Total	$9.023	$532,357	100%

Figure 4.7: Data (shades of black) and non-data (red) pixels.

shows how grid lines in a graph may be eliminated without any information loss. They are also considered to be the most prevalent form of distracting non-data pixel in dashboards.

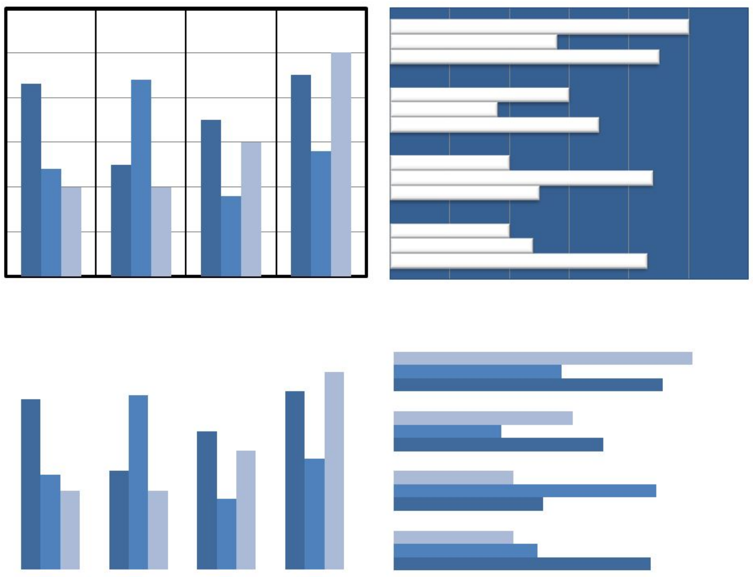


Figure 4.8: Effect of grid lines elimination.

While emphasizing important data, not only colors and contrast should be taken into an account, but also the location of the elements. Figure 4.9 shows different degrees of visual emphasis associated with different regions of a dashboard.

4.4 Bad design examples

According to Few, there might be distinguished several groups of reasons which lead to a poor usability of a dashboard:

1. *Exceeding the boundaries of a single screen.* It is important to show all information on a single screen, without scrolling or switching between multiple screens needed. This might help to discover some connections between different measures show. And on the contrary – it might lead to losing general picture or important details which were not presented on the first visible screen. The reason beyond this is that people can hold only small part of the information in their short-term memory. It is important to mention here that allowing navigation to further details may be useful for some dashboards, but it must be implemented carefully.

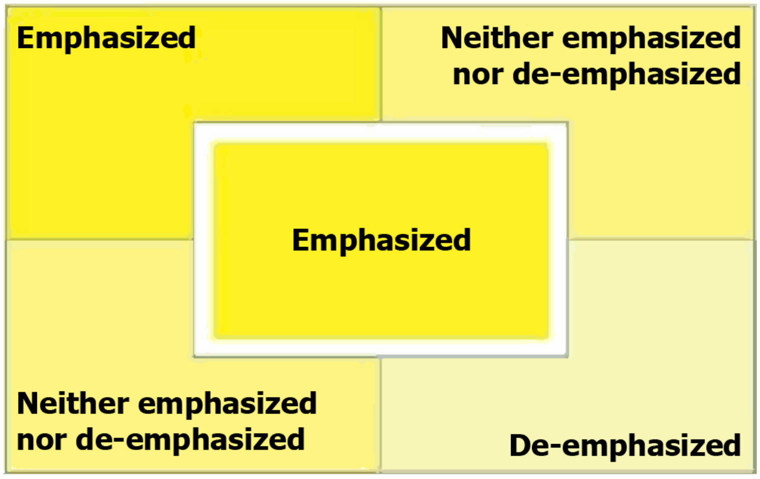

Emphasized
Neither emphasized nor de-emphasized
Emphasized
Neither emphasized nor de-emphasized
De-emphasized

Figure 4.9: Degrees of visual emphasis associated with different regions of a dashboard.

2. *Supplying inadequate context for the data.* Supplying context for measures means involving comparison to other similar measures or presenting several contexts. On the other hand, incorporating of a rich context may overwhelm a user with too much irrelevant data, so it is important to keep a balance here. The amount of a context presented in a dashboard also depends on its purpose and end users. More context should be provided only if it results in a real value.

3. *Displaying excessive details or precision.* Too precise details (for example $2,978,765.99 rather than $2,978,765 or $2,9M) can slow the process of perceiving the information.

4. *Choosing inappropriate display media.* This one is considered to be the most common design mistake, not only when speaking about dashboards, but also in other forms of quantitive data presentation. [8]. For example, using a pie chart when a bar chart would be more explanatory or using a graph when a table would work better.

5. *Introducing meaningless variety.* Since the main purpose of a dashboard is to present meaningful information, it is not a problem if all of it would be displayed with the help of the same type of graph. It would be a problem though if a variety of graphs would be used when their types are not adding any value. Also, consistency in the means of the display can help users to use similar patterns of information recognition. Explanation of how such patterns work was provided in Chapter 4 of this Thesis.

6. *Using poorly design display media.* It is important to use not only the correct type of a graph but also to design it properly. More information on the design of the different means of media display was provided in Chapter 4.

7. *Ineffective highlighting of important data.* The most important or critical information should always catch user's eye firstly. To avoid cluttering of the display useless decoration shouldn't be used at all. Another important rule is that color shouldn't be used haphazardly. Hot colors should be used to the elements which should catch an attention. Cooler colors are less noticeable. Similar colors may be used to create recognition patterns.

8. *Designing of unattractive visual display.* Unnecessary decoration elements shouldn't be used, as it was stated before. But, data still need to be presented in a nice way. If a dashboard would be ugly, the user would avoid to use it often. So aesthetics should be taken into account. These items show main issues which should be avoided while creating a dashboard. So it is important to keep them in mind while creating a new software for this purpose.

4.5 Analysis of Information Dashboards Usability

As shown in the previous chapters, information systems are becoming more and more widely used nowadays, even though before they were mainly used by corporations only. But still, quite often dashboards are not being used up to their full potential. The most common reason for failure is poorly designed implementation, and not the software itself. A lot of software developers are paying more attention to entertaining a customer with their product, and as a result, these displays may become annoying and not useful at all. An effective dashboard is a product not of cute gauges meters and traffic lights, but rather of informed design: more science than art, more simplicity than dazzle [8]. But since some users are getting used to such products, they do not realize that there might be a way to create a dashboard, which makes information perceiving a way easier. Cooper calls such effect *dancing bearware*. He says that the sad thing about dancing bearware is that most people are quite satisfied with the lumbering beast. Only when they see some software with a hight usability level, they begin to suspect that there is the world beyond suffering. So few software-based products have exhibited any real dancing ability that most people are honestly unaware that things could be better [5].

Stephen Few shows in his book [8] that there are many dashboards with usability problems. The main problem while creating a dashboard is to squeeze all relevant information into a one screen and to keep it easily understandable at the same time. But, it is often quite a challenging task to follow them, if one is not a design guru. So, there is a need for a tool, which helps to evaluate dashboards usability and quality, as well as classify them into groups (Figure 4.10).

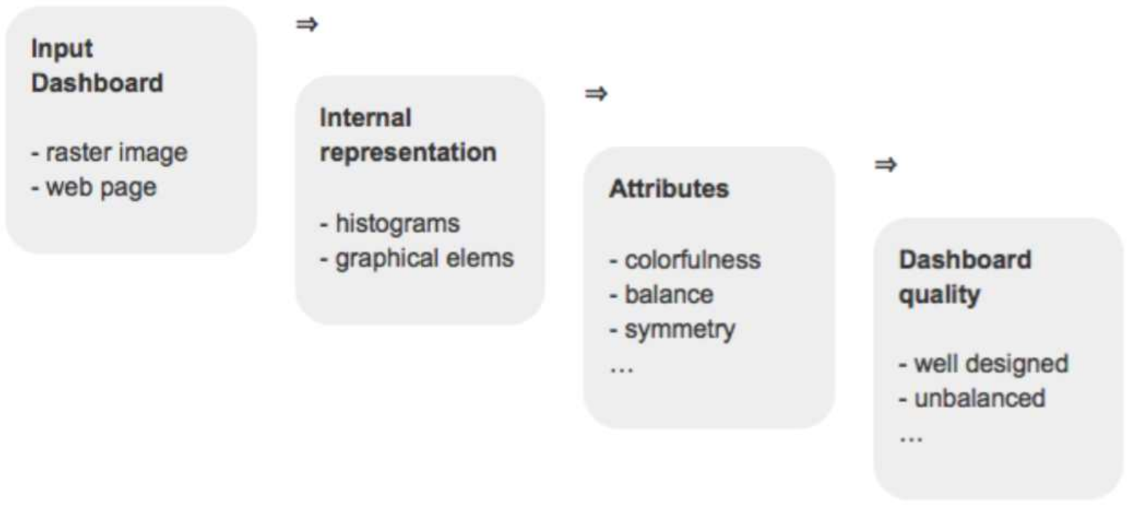


Figure 4.10: Steps of evaluating a dashboard usability.

There have already been developed a framework, which describes main steps of a dashboard evaluation. To make it work, there is a need in a big amount of test data. That can be weather raster image (matrix of pixels), or some language based dashboard, for example, HTML + CSS. Then, information about separate graphical elements, together with general dashboard histograms needs to be converted into an internal representation. Currently, for the internal representation XML is chosen. The purpose of this step is to select only the important information for the analysis of dashboard. The example of such XML representation is shown below.

```
<dashboard>
   <x>0</x>
   <y>0</y>
   <width>1908</width>
   <height>1148</height>
   <type>CHART</type>
   <graphicalElement>
      <x>0</x>
      <y>50</y>
      <width>52</width>
      <height>1118</height>
      <type>TOOLBAR</type>
   </graphicalElement>
</dashboard>
```

After such representation was created, it can then measure attributes which correspond with dashboard usability. The result of these calculations may be then used for the dashboard clarification and evaluation. Among attributes which are proposed for the evaluation are:

- data-pixel ratio,
- contrast,
- hue-saturation ratio,
- clear grid system.

The main purpose of this Thesis is to develop a software tool which can help to create dashboards easily so that they can be used as test data for machine learning algorithms.

Chapter 5

Analysis of available web technologies

Nowadays there are a lot of different ways how to build the web application. The technology stack is a combination of software products and programming languages used to create the web or mobile application. Since applications have two software components: client-side and server-side, there are front-end and back-end stacks respectively. Figure 5.1 shows an explanation of how such stacks work.

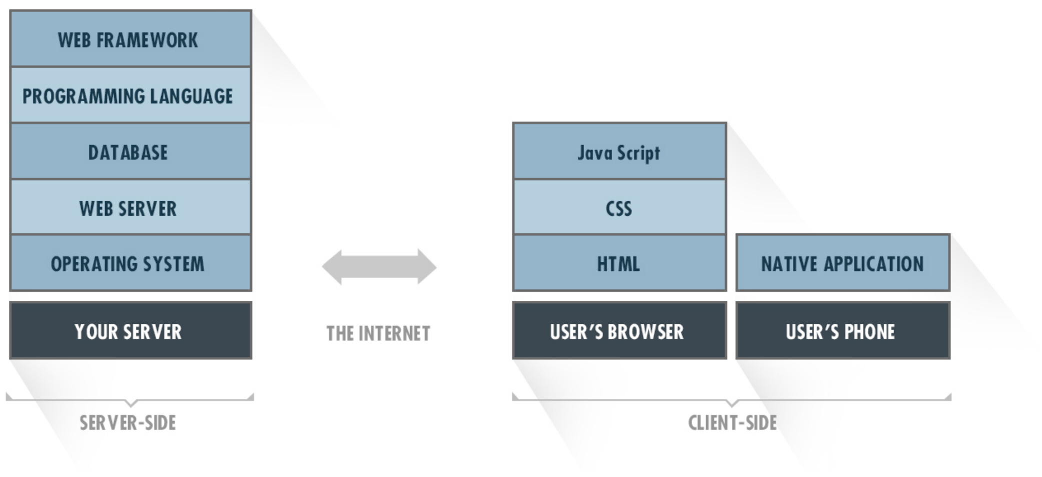


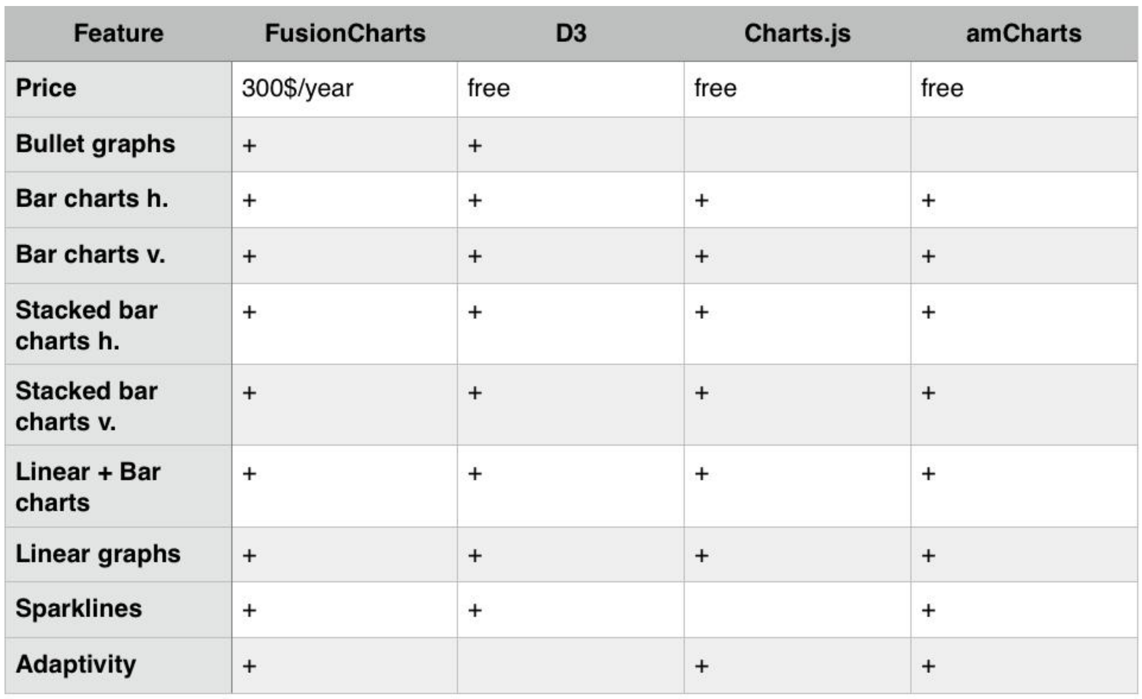
Figure 5.1: Server and client-side stacks.

It is called a stack because each layer of the application is built on the features of the previous one. The back-end is all about the logic behind the application, which will never be accessed directly by the user because all information is passed to front-end means firstly. And front-end is the visual part of the application, and the one user would see and interact with. This chapter shows a review of existing technologies, as well as the selection of the ones which would be used for the application.

5.1 Existing technologies review

Since the most important part of the application is going to be a dashboard creation pane together with single widgets, it is more appropriate to start from choosing a front-end technology stack. As the main front-end language, JavaScript (js) was selected because it provides needed dynamics. First of all, analysis of existing libraries for working with graphs was performed. Since this list is quite a long one, the first step is a selection of the main JavaScript framework for a front-end representation.

For the development of an application for fast and easy dashboard creation, Vue.js framework was selected as the most suitable one. It is open-source and supports the development of reactive components for Modern Web interfaces. It uses model – view – View-Model (MVVM) paradigm, which helped in the simplification of the design. It was selected because it helps easily create re-usable components, which would have the same HTML structure but still would use different parameters for styling. This feature is extremely useful while developing the library of elements for a dashboard, described in Section 4.2. Another useful for this particular project feature of Vue.js is that it uses two-way data binding to update the model and view, which allows reactivity of the web application. Then, after the main framework is selected, the list of possible js libraries for working with graphs had shortened. To select from them, following criteria were developed: free/paid, all needed elements available, adaptivity. The result is shown in the next table:



Feature	FusionCharts	D3	Charts.js	amCharts
Price	300$/year	free	free	free
Bullet graphs	+	+		
Bar charts h.	+	+	+	+
Bar charts v.	+	+	+	+
Stacked bar charts h.	+	+	+	+
Stacked bar charts v.	+	+	+	+
Linear + Bar charts	+	+	+	+
Linear graphs	+	+	+	+
Sparklines	+	+		+
Adaptivity	+		+	+

Figure 5.2: Comparision of JavaScript libraries for charts and graphs.

As a result of this research, D3.js library was chosen for the development. There also exist a several open source D3-based libraries for building custom charts and graphs, for example – xCharts. This one may be used to scale application to other custom elements. XCharts library also would help to ensure that all graphs are working good when viewed from a mobile device.

At the beginning of each project it is important to set up *version control system*. Git is selected for this particular project. It allows remote repositories creation so that working from several computers would be easier. Also, useful feature of git is that it has branching system, so that when a developer needs to try out a new feature, they can create a new branch in the repository and work there without affecting the code in the main branch. It is merged with Master branch only after tests.

5.2 Mobile usability

According to a survey conducted by Harris Interactive for Tealeaf[1], 23% of users were cursing at their phones, 11% have screamed at them, and 4% have actually thrown their phone when experiencing a problem with online interaction. This shows increasing need of mobile optimization for web applications. The problem here is that nowadays mobile devices come in all resolutions and screen sizes, so there is no clear resolution or cutoff to maintain.

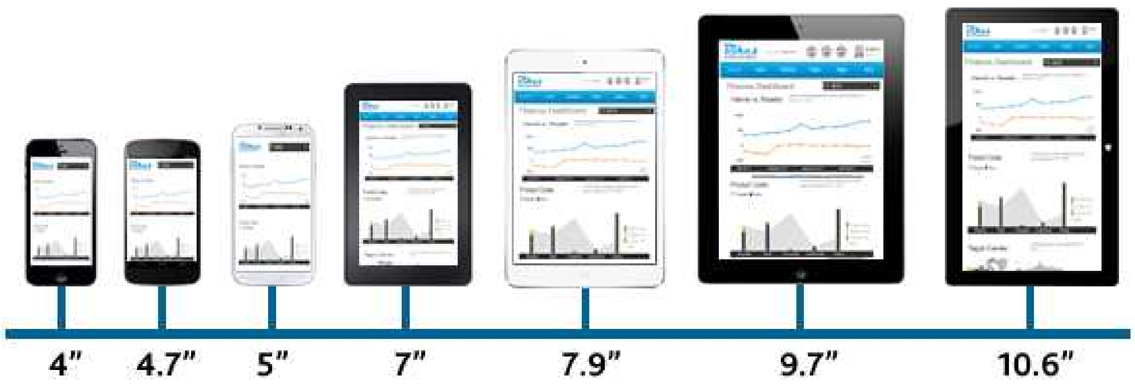


Figure 5.3: Different screen sizes which had to be taken into account while adaptive development.[2]

The best way to make a web application looking good on different devices is to make it responsive. Nowadays, there are several frameworks for this purpose, along with manual CSS media queries. For this project, Bootstrap Framework is chosen as the mean of responsiveness because it has clear 12-columns grid structure and supports a principle *mobile-first*. This is important because it is always more difficult to shrink information to the size of a mobile device then vice-versa.

[1]Mobile transactions survey http://www.marketwired.com/press-release/tealeaf-announces-new-mobile-transaction-research-conducted-harris-interactive-shows-1419058.htm

[2]Source: XappMedia

Chapter 6

UXgraph library

Since Vue.js was selected as a front-end base for the application, there was a need in the Vue extension for working with D3. Developing graphs in plain D3 is also possible, but it offers a lot of possibilities and it takes a lot of efforts to achieve needed design quality. Currently, there already exist several Vue.js / D3.js open-source components available on the internet. But firstly, they do not represent even minimum set of needed graph types and allow you to create only one or two types. For example, in Tyrone Tudenhope GitHub project[1] provides only sparklines and line charts. There are a lot of different D3 examples created by Mike Bostock[2], but they do not offer a possibility to quickly set custom settings. The other problem of their components (and the main problem solved in this project) is that they do not fulfill design requirements specified by different graphic visualization experts, Stephen Few [8] in particular. Not following these requirements makes these graphs difficult to understand, and as a result – less effective. At the current stage, UXgraph also contains only several types of graphs, but they all already follow required design principles and allow easy definition of custom properties, so the first part of the challenge was solved. The extension is already published in its beta version on npm (*Node Project Management*)[3].

6.1 UXgraph implementation

UXgraph requires running Vue application or vue.js script connected to any other codebase. Vue.js is a progressive framework for building user interfaces. It is focused on the View Layer, and provides reactive and composable view components. Vue advantage is that even though it is a JavaScript framework, it still supports HTML and CSS right in the single file components (on the contrary of React, where everything is just JS).

UXgraph supports latest versions of Vue and D3. This feature is quite important since with the last version change there were some core updates in both of them.[4] [5]

[1] `https://github.com/johnnynotsolucky/samples/tree/master/vuejs-d3`
[2] `https://bl.ocks.org/mbostock`
[3] `https://www.npmjs.com/package/ux-graph`
[4] Vue2.x migration guide
[5] D3v.4 changes description

6.2 Vue reusable components

Components in Vue can help to extend basic HTML elements to encapsulate reusable code. To make it work, Vue's compiler attaches special behavior to these custom elements. To register a global UXgraph component following steps are needed:

```
Vue.component('my-component', {
  // options
})
```

After that, the component may be used in the web page template as a custom element:

```
<my-component></my-component>
```

Every graph type component is located in its own single file template, which contains HTML, scripts for declaring properties and behavior and styling (Figure 6.1). That makes it possible to use them all independently. Following components are already implemented and can be imported: *Sparklines*, *Linecharts*, *Barcharts*, *HorizontalBarcharts*. *StackedBarcharts* and *BulletGraphs* components would be added during the next stage.

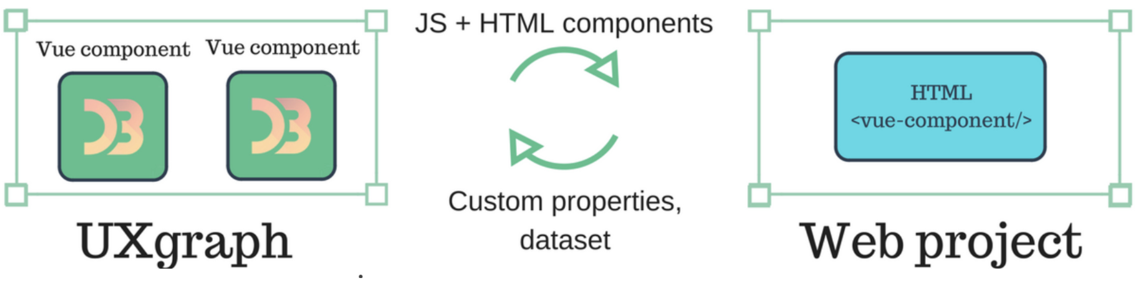


Figure 6.1: Integration of UXgraph into a Web project

General D3 methods are connected to .vue components by installing `d3` package via NPM, and then importing `d3` as a global variable. But to make application reactive, it is not enough to create D3 SVG elements as usual. That's why UXgraph uses Vue mounted event to call a method for creating a graph after a component instance was rendered. When calling the method it is also needed to specify its parameters, which represent customizable graph settings. For example, for sparklines it would be:

```
mounted () {
    this.createSparkline
    ('#id', this.data, this.label,
    this.circle, this.color)
},
methods: {
    createSparkline(id, data, label,
    circle, color) {}
}
```

As a result, variables represented by these parameters may be used anywhere during the SVG construction.

6.3 Passing data between components

Another Vue feature which is used in UXgraph is a possibility to pass data to a component with properties. Since components are reusable and can be inserted in basically any place of a web page, it is important to keep them in their own isolated scope, and so – not to directly reference parent data from a child component. That's why all data in this library are passed to child components using props. Firstly, in all child components (which contain graph templates) properties are defined using Vue *props* option:

```
Vue.component('sparkline', {
  // props declaration
  props: ['data', 'color',
  'circle', 'label']
})
```

After this, every property can be referred as this.propertyName within a component. Then properties can be passed from the parent template like following:

```
<sparkline
    color="#4682B4"
    label="Daily defects"
    circle=true>
</sparkline>
```

Result is shown on Figure 6.2.

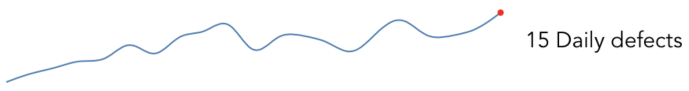


Figure 6.2: Example of generated sparkline component, with custom parameters

For all UXgraph components default properties are specified in order to make including of a new component easy (Figure 6.3). Users need to specify a property only if they want to use custom values.

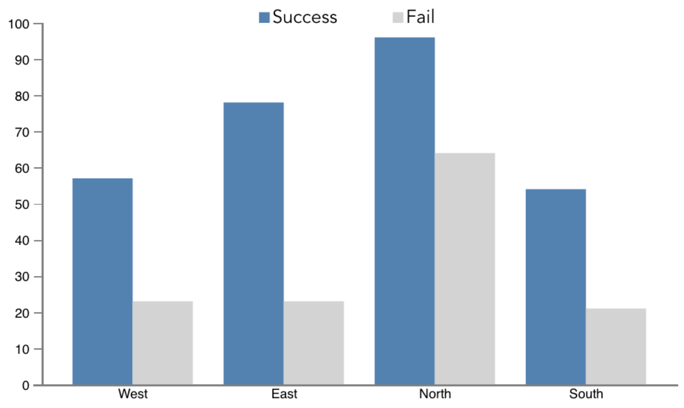


Figure 6.3: Example of a bar chart component included without any parameters

Chapter 7

Web application for information dashboard design

The main goal of this project is to create a web application which would allow researchers to quickly and simply create dashboards, all components of which fulfill best UX requirements, described in Chapter 4.

7.1 User and Task Analysis

Task analysis is the process of learning about ordinary users by observing them in action to understand in detail how they perform their tasks and achieve their intended goals. It is really important to make task analysis as early as possible in every project, as it helps you to understand better goals and needs of the users, their actions and experiences, physical environment and previous knowledge influence.

There are several types of task analysis [1]. *Cognitive Task Analysis* is required when decision-making, problem-solving, memory and attention of users are involved in the project. And *Hierarchical Task Analysis* is focused on decomposing high-level tasks into several layers of subtasks. But to create full task analysis it is important to perform user analysis firstly. Neither User Analysis or Task Analysis are about the interface directly, together they help to understand the constraints and requirements for future interfaces.

7.1.1 Context

Primary users are already interested in usage of the application. They are researchers, which need to generate big amounts of informational dashboards quickly for testing purposes. Moreover, they do not want to spend time with every single widget, making all design adjustments. They need to have all UX settings by default. They would work with the application in their offices or labs, and during working hours mostly. While interacting with the application users probably wouldn't be interrupted a lot, as they are in the working environment. But, since the creation of the dashboard itself is not their primary goal, they probably wouldn't like to spend too much time on this task, or on learning about how to use the application.

7.1.2 Design guidelines

Based on human psychology design guidelines were developed. This is not something that changes as fashion does, but design patterns which correlate with how our brain works. For this web application design guidelines described by Nielsen and Molich in 1990 [1] were chosen, as the ones which describe needed system more accurately:

- consistency and standards,
- visibility of system status,
- match between system and real world,
- user control and freedom,
- error prevention,
- recognition rather than recall,
- aesthetic and minimalist design,
- provide online documentation and help.

In order to achieve goals of the project and make it successful, following guidelines are added:

Design for short sessions:

- Avoid long startup
- Avoid long configurations

Design to make dashboard creation quick:

- Easy way to add a new sample
- Default settings which follow best UX practices
- Several sets of designs available
- Test data exists and easy to fill in

Design for researchers:

- Ability to autogenerate testing samples
- Saving history of interaction
- Export of samples in needed format

Design to make application scalable:

- Responsive design
- Cross-platform solution
- Content, optimized for future usage on tablets
- Some popular platform

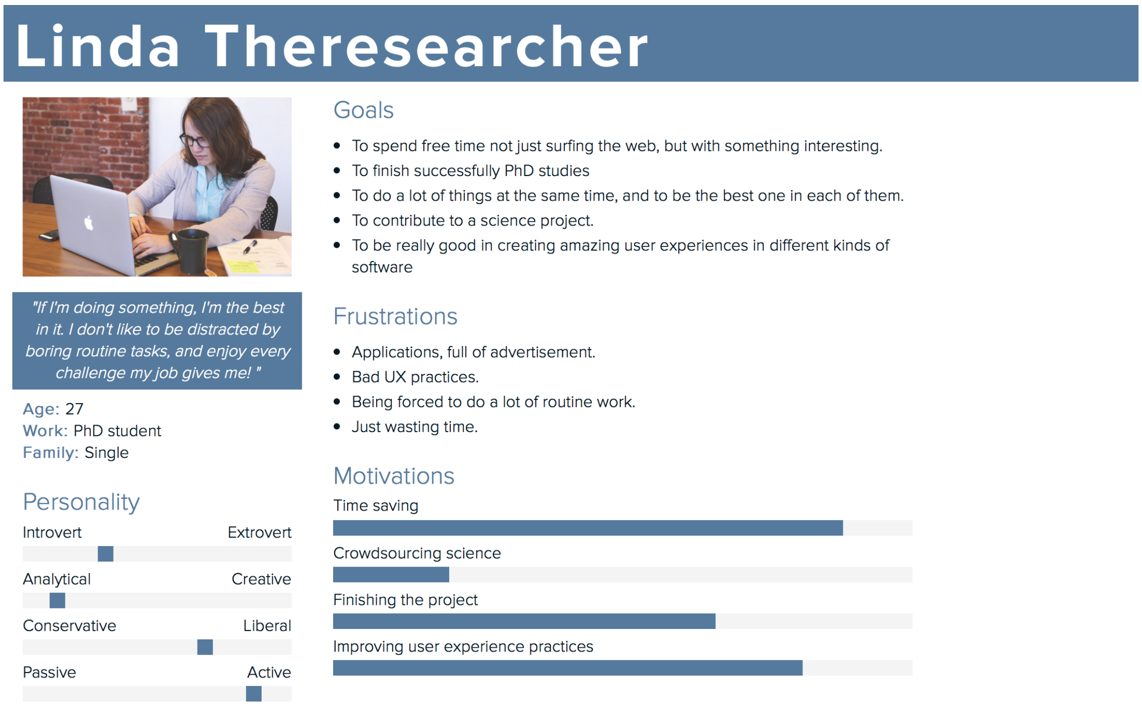
As a design guideline for future application, Material Design is chosen.

7.1.3 Primary persona

While working on any application it is important for a developer to keep in mind that "You are not the user". In other words, it is important to understand the needs of the end user. It is then possible to identify the features and functionality that will make the intranet or website a success, and how the design can support users with different goals and levels of skill. There are many ways to identify the needs of users, such as usability testing, interviewing users, discussions with business stakeholders, and conducting surveys. However, one technique that has grown in popularity and acceptance is the use of personas: the development of archetypal users to direct the vision and design of a web solution.

Personas identify the user motivations, expectations and goals responsible for driving online behavior, and bring users to life by giving them names, personalities and often a photo. Although personas are fictitious, they are based on knowledge of real users. The goal is to find a single persona from the set whose needs and goals can be completely and happily satisfied by a single interface without disenfranchising any of the other personas. This persona is typically referred to as the primary persona. By definition, each primary persona requires their own user interface in a particular application. As in my application there is going to be only one primary persona (the researcher who is creating dashboards for test purposes), there should be only one interface also. Figure 7.1 shows my suggestion of primary persona for this project.

Within this project, only one abstract user may be defined – researcher. Since the application is developed directly for their purposes, the main interface should be adjusted accordingly. So, the user is aware of the goals of the project, is motivated enough to use the application and looks for a way to generate a sample for testing.



Linda Theresearcher

"If I'm doing something, I'm the best in it. I don't like to be distracted by boring routine tasks, and enjoy every challenge my job gives me! "

Age: 27
Work: PhD student
Family: Single

Goals

- To spend free time not just surfing the web, but with something interesting.
- To finish successfully PhD studies
- To do a lot of things at the same time, and to be the best one in each of them.
- To contribute to a science project.
- To be really good in creating amazing user experiences in different kinds of software

Frustrations

- Applications, full of advertisement.
- Bad UX practices.
- Being forced to do a lot of routine work.
- Just wasting time.

Figure 7.1: Primary persona.
Source of portrait picture: Max Pixel

7.2 Features

Application at its current form will not be suitable for implementing an information dashboard for a real business since its backend will not be suitable for this. But, it will be easily scalable, so no problems in adding such a function. Every researcher would need to create an account, in order to save created sets of widgets, preferences and etc. After login, they get a possibility to work with the app.

- User needs to create an account and log in to use the application,
- Landing page of the application shows quick buttons for creating different predefined dashboards,
- User can save created dashboards,
- User can start generating script, to generate big amount of dashboards, which satisfy certain criteria,
- User can leave comments under generated samples (for research purposes),
- User can export a dashboard.

7.3 Technology stack

Based on the research described in Chapter 4, following technology stack is defined for the web application (Figure 7.2).

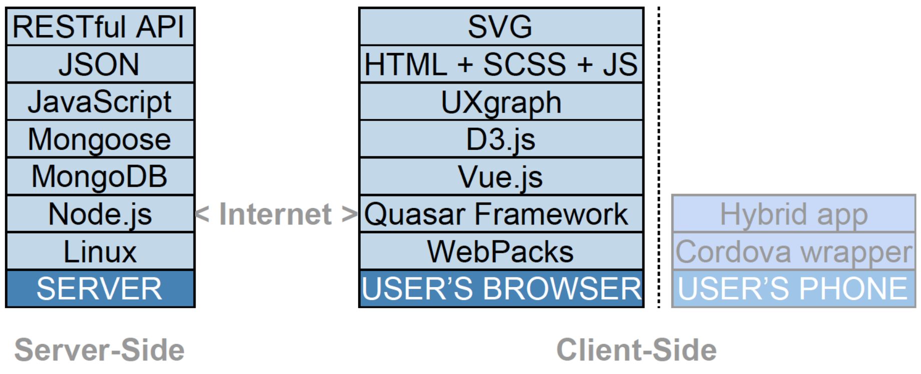


Figure 7.2: Illustration of technologies used for server and client sides of the application, and explanation how hybrid mobile application may be generated in the next stage.

Frontend:

- **Bootstrap**. It is one of the most popular frameworks for developing responsive ad mobile first projects, which is a very important feature in regards to this project. It is based on HTML, CSS, JS.
- **jQuery**. It is cross-platform JavaScript library, which is aimed to handle events on HTML pages. Such widgets as a tooltip (popup tips which make interactions with the application easier for the user), slider, accordion (displays collapsible content panels for presenting information in a limited amount of space) and date picker (allows to select a date from a popup or inline calendar) might be used in the project.

- **HTML5**. A markup language used for structuring and presenting content on the World Wide Web. In HTML5 version a lot of useful for the project features were added. Mainly, the ones which are connected to multimedia and graphical content. As, for example, `<video>` element, which helps to include and manage video content on the page without any additional plugins or APIs. Also, new HTML5 elements, as `<main>`, `<header>`, `<footer>`, `<nav>` are helping to improve semantic structure of the document.
- **CSS3**. It is the latest standard for CSS stylesheet language which is going to be used to describe the presentation of the HTML document.
- **Vue.js**. It is a library for building interactive web interfaces. It provides data-reactive components with a simple and flexible API, and is component-oriented, what is important for the current project because of a need in reusable components for the widgets.
- **D3.js**. A JavaScript library for visualizing data with HTML, SVG, and CSS. It will be used for building charts and graphs.
- **gridstack.js**. A jQuery plugin for creating dashboard grid. Allows drag-n-drop and changing size of the widgets.

Backend:

- **OS**: Linux Ubuntu. To serve all needed purposes version should be at least Ubuntu 16.01.
- **Web server**: Node.js HTTP server
- **Database**: MongoDB

Build system:

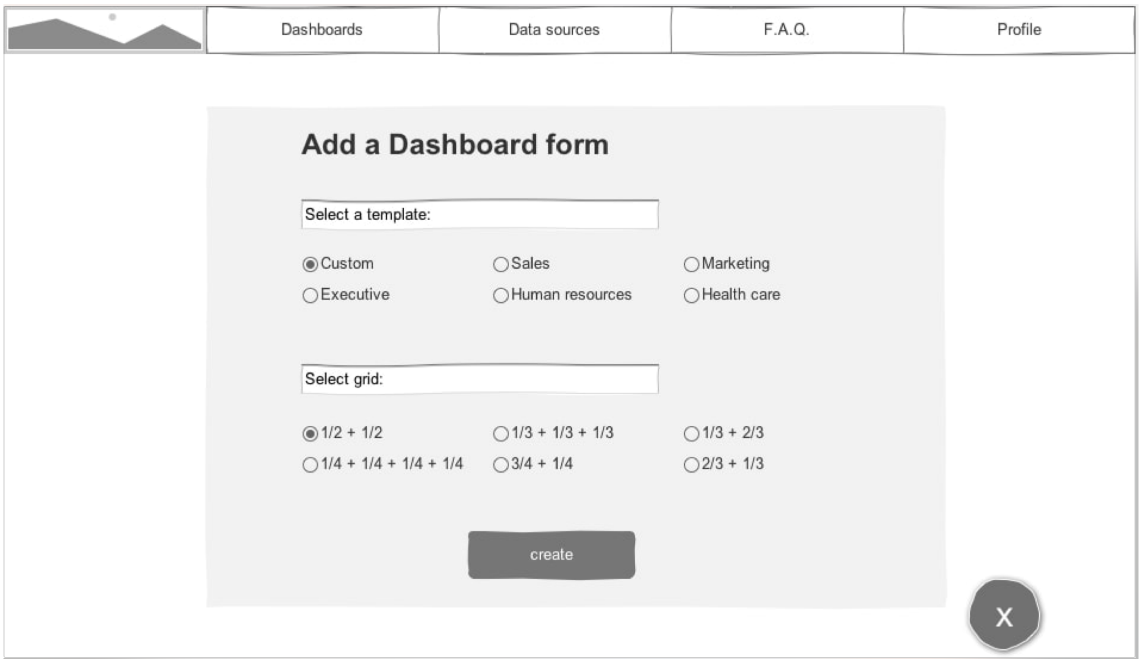
- **Gulp.js**. This is a toolkit for automating tasks. In the project, gulp would be used to compile sass and to minify resulting CSS, to combine and minify all external JavaScript files and to optimize images. It also makes the development process easier by enabling CSS and js spelling checker and browser live reload functionality.

For convenient versioning control git repository would be created.

7.4 Application prototype

Application prototype is developed based on requirements described throughout the paper using Axure software. This is a software for prototyping of applications. It should be used before writing any code, in order to validate a solution or make any user tests. Page structure is as follows:

1. Landing page. Explains the project in several words, has the login form.
2. Add a dashboard (Figure 7.3). This is basically a pop-up form, which asynchronously loads after any content when floating button is pressed. After all options are selected – a grid for constructing a dashboard is shown.



Add a Dashboard form

Select a template:

(● Custom) ○ Sales ○ Marketing
○ Executive ○ Human resources ○ Health care

Select grid:

(● 1/2 + 1/2) ○ 1/3 + 1/3 + 1/3 ○ 1/3 + 2/3
○ 1/4 + 1/4 + 1/4 + 1/4 ○ 3/4 + 1/4 ○ 2/3 + 1/3

Figure 7.3: Add a dashboard form.

3. Constructing a dashboard page (Figure 7.4). Contains a selection of possible widgets, and selected grid system. The user can add needed widgets into selected grid cells.
4. List of all dashboards created by users (Figure 7.5). With buttons for edit, export or delete a dashboard.
5. Profile page. Where all information about logged in user is listed.
6. FAQ page. Contains general information about the application and instructions on functionality.
7. Data sources. Lists possible data sources and have a possibility to import new ones or create them manually.

7.5 Application architecture

The Project uses Single-Page Application (SPA) architecture. It loads a single HTML page and dynamically updates it as a user interacts with the app. So, View Layer is handled by Vue, with routing managed by vue-router. Server communication works with the help of vue-resource, plugin for interfacing with a RESTful backend. And everything is built with Webpack build tool.

The base for a front-end of this application is Vue, with connected UXgraph components. Graphs are generated fully on a client side since dataset and other parameters are set as Vue properties.

7.5.1 RESTful API for handling dashboards operations

An important feature is that all back-end logic is made and hosted as a separate Node.js project, which may be accessed through API. Since front-end and back-end are not closely

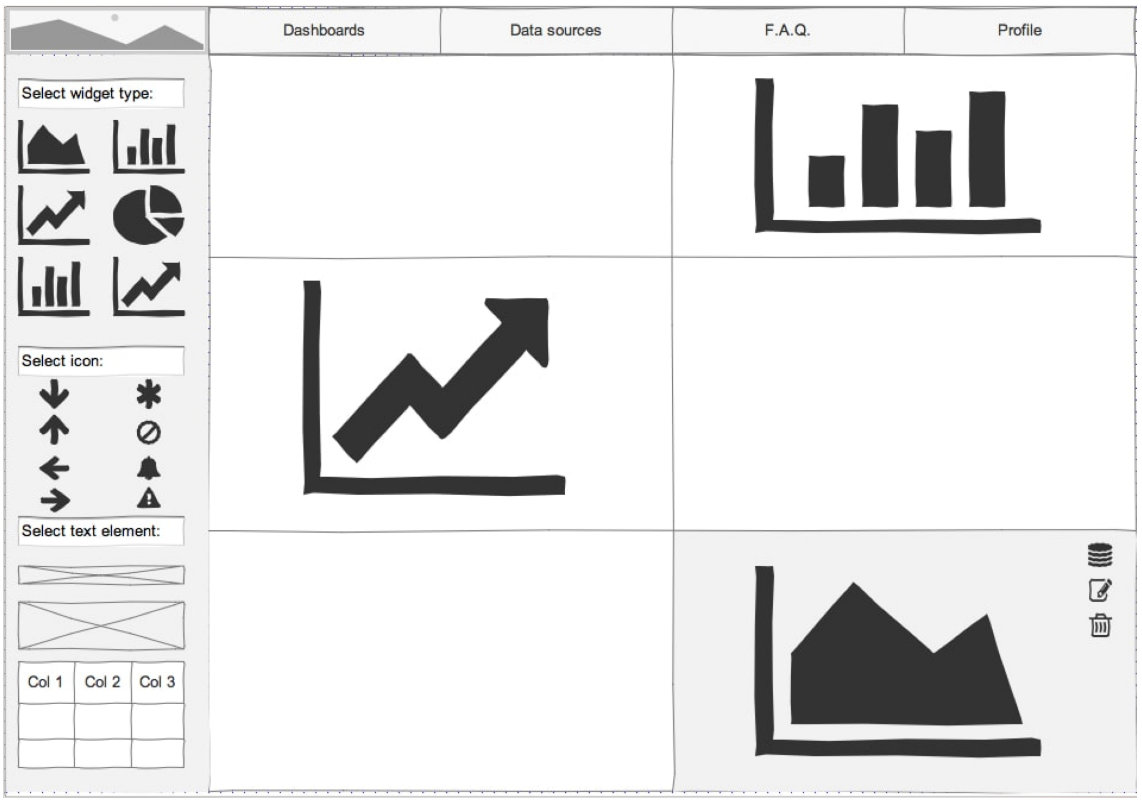
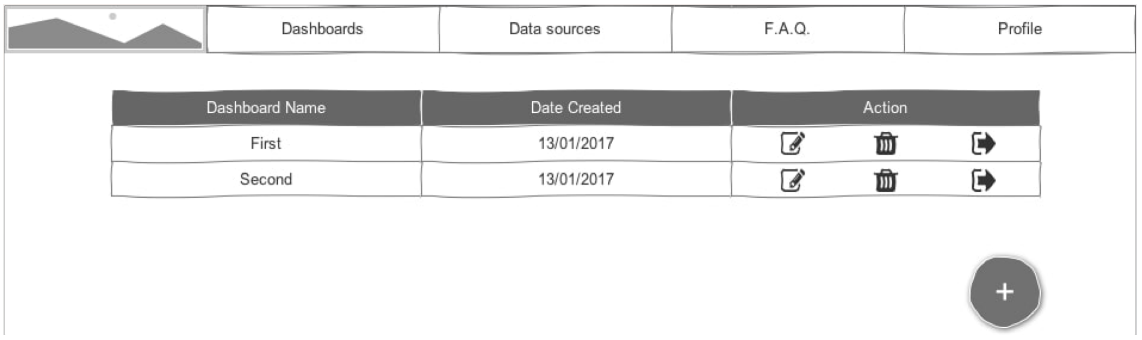


Figure 7.4: Constructing a dashboard page.



Dashboard Name	Date Created	Action
First	13/01/2017	✎ 🗑 ➜
Second	13/01/2017	✎ 🗑 ➜

Figure 7.5: List of all dashboards.

integrated together, it is possible to add or modify backend operations, without influencing front-end and vice verse.

MongoDB is selected since it works with objects saved as JSON, what makes it easy to export them later on [12]. Plus, it is possible to save objects from `gridstack.js` (used to enable drag-n-drop for widgets construction pane) directly as Mongo JSON objects. All collaboration with MongoDB is managed with the help of Mongoose. This is a MongoDB object modeling tool designed to work in an asynchronous environment.

7.5.2 Dynamic props

One of the tasks was to update graphs dynamically, whenever their settings are changed. For this, another Vue function is used: *v-bind*. It allows to dynamically bind props to certain data on a parent. [16] So, when the user selects a new color for a graph, this parameter dynamically flows down to the child and it is updated on the graph without page reload. Example usage is:

```
<input v-model="parentColor">
<sparkline
    data="[3,2,1,4]"
    v-bind:color="parentColor">
</sparkline>
```

To redraw graphs when a property is changed without page reload, there were developed custom watch functions. Every watcher is triggered when a specific property is changed, and then it calls a method which is responsible for drawing the SVG.

7.5.3 Explanation of application deploying process

Since the application consists of two separate parts – back end and front end – it is needed to deploy both of them separately. Firstly server side has to be set up. After the repository is downloaded, there are only several steps to take. The application is using Node.js as a base for the backend, so all needed packages (libraries) are gathered in the package.json file. That file also contains minimum required version for each package. They can be all installed at once with the help of npm (Node Project Manager) `install` command. This command also installs all sub-dependencies for every package and takes care of specific OS requirements. Before starting Node server, it is needed to ensure that MongoDB is installed and running on the machine. This covers API deployment.

The frontend is built with Quasar Framework, which has two options: development mode and a build version. While development it supplies its own server, but to deploy live website it is needed to serve build files to a server.

More detailed instructions can be found in the README files in the GitHub repositories.

Chapter 8

User testing and evaluation

All process of this application development was iterative. That means that the phases «design – test – measure – redesign» were repeated several times. It helps to discover mistakes at early stages, and fix them immediately [1]. So, there were two kinds of testing: *formative user testing*, performed at the prototyping stage, and *summative usability testing*.

8.1 Primary functionality

Resulting application has all functionality covered in the previous chapters.

- **Create new dashboard.**

  As soon as a users are logged in in the application, they see the list of all available dashboards, and a button allowing to create a new one. This page is build based on the prototype (Figure 7.5). The new functionality here is a possibility to select a custom grid (then the user receives blank canvas for designing a new dashboard), or one of the tree predefined grids. This option makes creating new dashboard faster. Figure 8.1 shows an example of a predefined dashboard.

- **Generate several dashboards.**

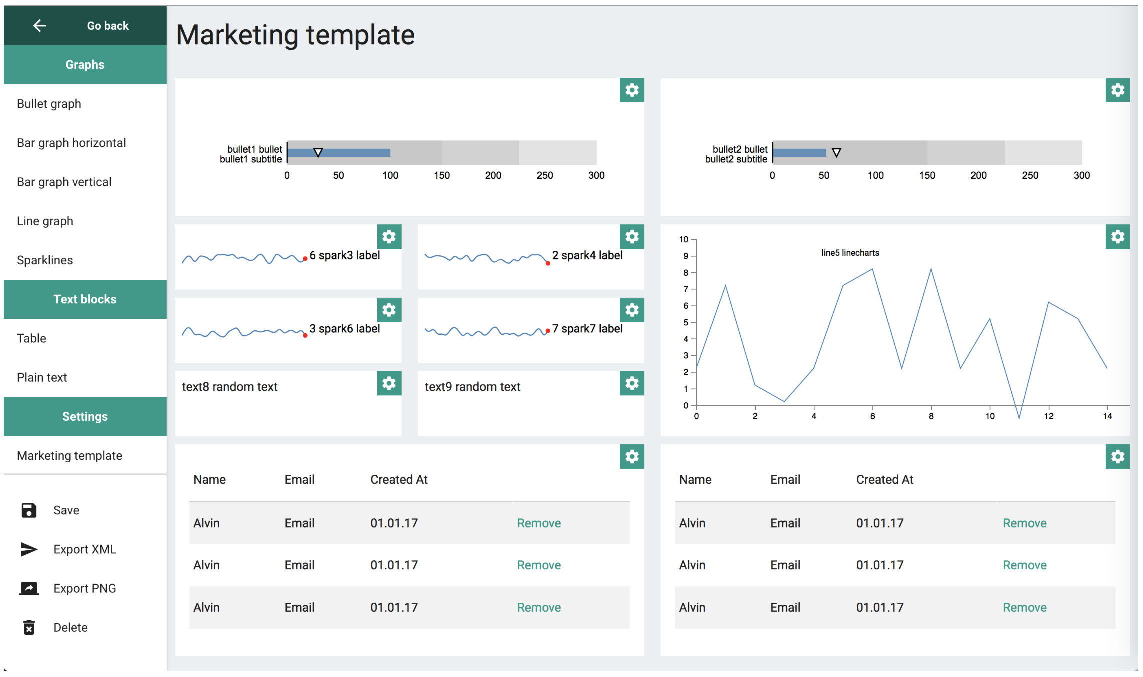
  To make dashboards creation process even faster, in the application exists a functionality to generate several dashboards at once. They would all have the same set of widgets, but generated datasets would be different. To do this, a user just needs to select amount of dashboards, and write a prefix for the titles. Each title would then consist of a prefix plus a number. In future, there is a possibility to extend the application, so that resulting set of dashboards would have more different parameters (like a color of widgets, for example).

- **Change grid or widget sizes.**

  All widgets are draggable and resizable (Figure 8.2). This allows quick manipulation with their sizes and positions directly on the canvas.

- **Change chart parameters and dataset.**

  Every widget has Settings layer (Figure 8.3). Settings are different for each widget, but for all of them, they are dynamic. This means that changes made in the Settings layer are displayed on the widget immediately, no need to refresh the page or press Save. It helps to make the process fast.



Marketing template

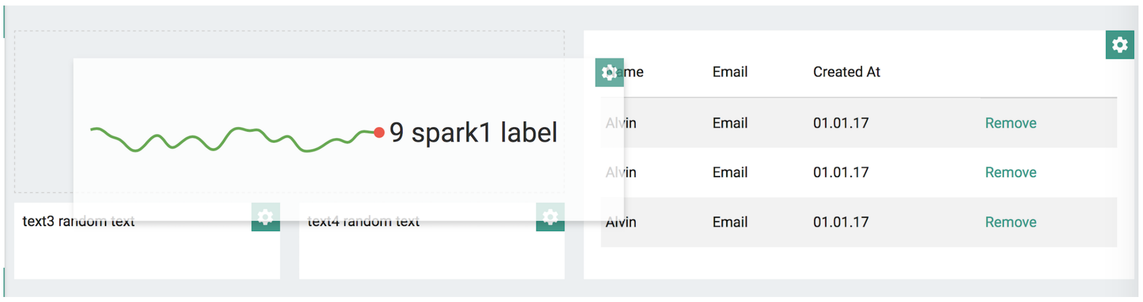
text8 random text

text9 random text

Name	Email	Created At	
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove

Name	Email	Created At	
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove

Figure 8.1: Predefined grid for a new dashboard creation.



text3 random text

text4 random text

Name	Email	Created At	
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove

Figure 8.2: Demonstration of drag-n-drop possibility. Grid cells available for a widget location are highlighted, and snap-to-grid functionality is enabled, which leads to a clean grid structure.

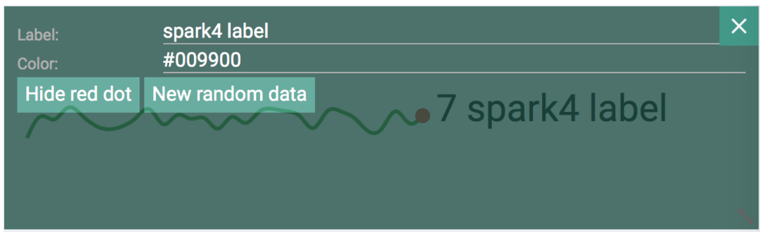

Label: spark4 label
Color: #009900

Figure 8.3: Sparklines settings layer.

- **Save dashboard, export to XML, export to PNG**

  Buttons for manipulation with dashboards in the database are located in the right sidebar, and always visible to a user. There are the following possibilities: Save, Delete, Export XML, Export PNG.

  Export to XML is developed specifically for this project, and resulting XML satisfies requirements needed for future analysis. An example of the XML is as follows:

```
<dashboard>
    <height>260</height>
    <width>1193</width>
    <title>Dashboard title</title>
    <graphicalElement>
        <height>260</height>
        <width>596</width>
        <y>0</y>
        <x>298</x>
        <type>linegraph</type>
        <style>
            <color>#990</color>
            <label>line2 linecharts</label>
            <dataset>[...]</dataset>
            <x>1</x>
            <y>0</y>
            <width>2</width>
            <height>3</height>
        </style>
    </graphicalElement>
    <graphicalElement>
        <height>80</height>
        <width>298</width>
        <y>0</y>
        <x>0</x>
        <type>sparklines</type>
        <style>
            <label>spark1 label</label>
            <redDot>true</redDot>
            <dataset>
            [...]
            </dataset>
            <color>#009900</color>
            <x>0</x>
            <y>0</y>
            <width>1</width>
            <height>1</height>
        </style>
    </graphicalElement>
</dashboard>
```

This example shows XML created for a dashboard containing two widgets only: Sparklines and Line Chart. Important feature is that every widget is saved as `<graphicalElement>`, and a set of the primary root settings (`height`, `width`, `x` and `y` coordinates, type) is the same for every widget type. Then settings which are different are gathered within the `<style>` part. This helps to use the same scripts for the analysis later on. The structure of the resulting XML may be easily adjusted. Since the application works with MongoDG all objects are saved as JSON, which is easily translated to any XML structure.

An example of the generated PNG file is shown on the Figure 8.4. PNG files are being generated using the library for Node.js, which takes specified DOM elements for the future image. This helps not to display in the PNG sidebar and settings icons.

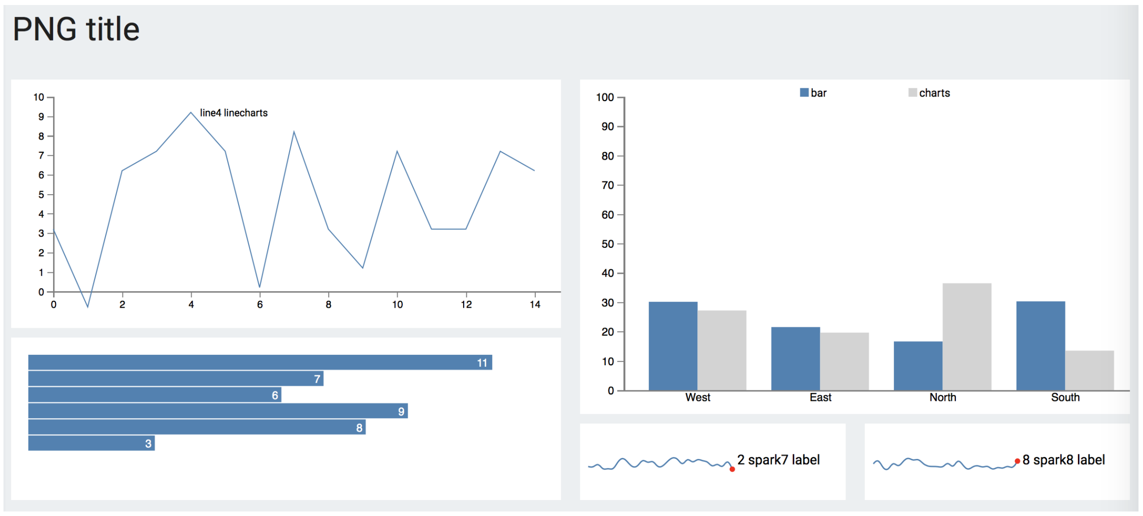


Figure 8.4: PNG file generated for the custom dashboard.

8.2 Formative usability testing

Formative usability testing should be performed at early design stages, or even better – at the prototype stage. It is aimed to show what parts of the interface aren't usable, or aren't intuitive enough . So it can take the role of a support tool for decision making [1]. In order to perform Formative usability testing, three users were selected. They were invited to interact with interactive application prototype build with Axure tools. They weren't given any specific tasks and had just to figure out how the application works, trying to press all buttons and menu items. Then they were asked to write down their observations, suggestions, and difficulties.

User 1: female, engineer, 50 years old

- Reaction: first reaction was a question «In what language F.A.Q. section is written? I can't understand it», referring to Lorem Ipsum text [1] usage in the answers. But in general she liked everything, everything was clear and comfortable.

[1] `http://www.lipsum.com`

- Suggestions: make it more clear how to delete the graphs; open generated image in a new tab, rather then in the same one.
- Difficulties: haven't discovered drag-n-drop possibility, so because of that didn't find a way to delete a widget from a dashboard.

User 2: male, psychology student, 23 years old

- Reaction: «I cannot find how to save a dashboard or exit the settings screen!»
- Suggestions: make some clear explanation about what each icon should do; create obvious navigation when already at the editing a dashboard screen, make buttons for saving or exporting always visible; make inputs the same width as checkboxes on the Add a dashboard modal window.
- Difficulties: the user couldn't find how to save the dashboards.

User 3: female, web developer, 30 years old

- Reaction: «It all looks good, I understand that at this stage graphs are not dynamic because it is difficult to achieve at the prototyping stage, but I'd really like to see how they change their settings when already live. Also, why does «select a template» label looks like a text input? This shouldn't happen in the live application»
- Suggestions: add more settings, not only color but also a label, title and etc.
- Difficulties: no difficulties found, the user knew general idea that this is only a prototype, and having a solid web development experience guessed all functionality out of it.

So, in general, all of them were satisfied with the suggested prototype. Some suggestions they made are covered in the design (like adding more settings, or making deleting functionality more obvious). Others are prototype limitations only. Some users also had troubles with understanding a dataset concept, but they are not in the target group and didn't have an experience of working with dashboards before. So since the aim was to satisfy needs of a primary persona in the first order, such issues wouldn't be addressed.

8.3 Summative usability evaluation

Summative usability evaluation should be performed when the design is almost finished, and main application functionality is ready. It tells you how convenient interface is, measured by values like time to complete the task, satisfaction score, etc. In other words, summative usability evaluation is a Quality Assurance (QA) type of tests.

Summative usability evaluation was performed on the already deployed website. For this task, users were given direct instructions and a questionnaire aimed to guide them through the main tasks. They were then asked to write down their suggestions and difficulties, also to describe how easily discoverable each functionality was.

Main tasks which user had to perform are basically the same guidelines, which were described in Section 7.2. All tasks are grouped within three parts: general application features, single widget related features and dashboards features. After finishing each of the sections, users were asked how satisfied in general they are with the interface. Results for each group are on the Figure 8.5, Figure 8.6 and Figure 8.7 respectively.

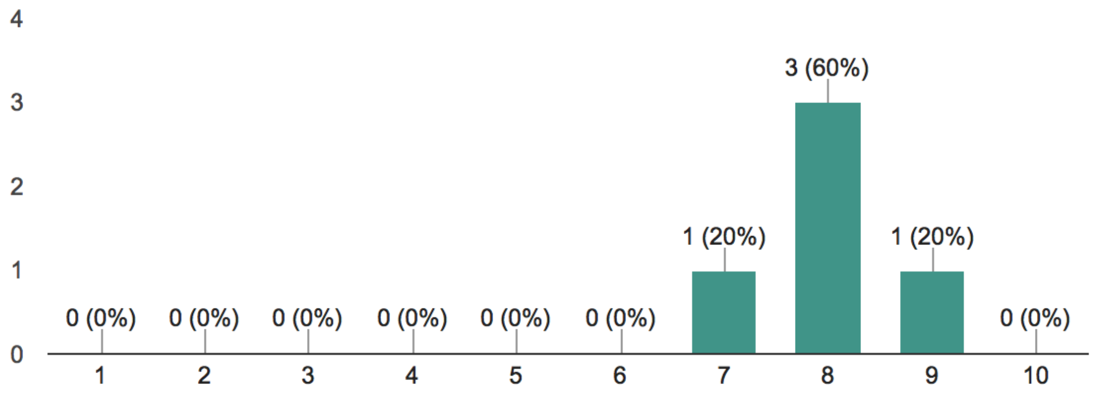


Figure 8.5: Usability evaluation results for actions realted with user account and login functionality.

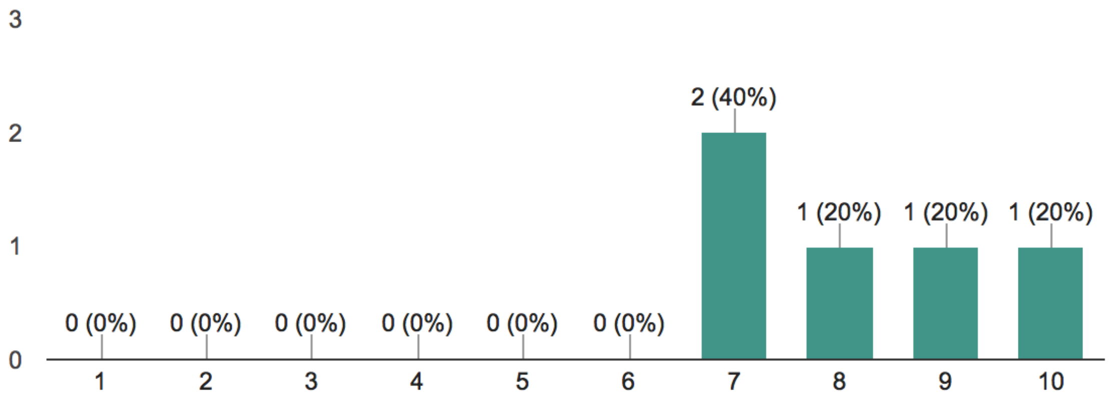


Figure 8.6: Usability evaluation results for manipulating a single widget.

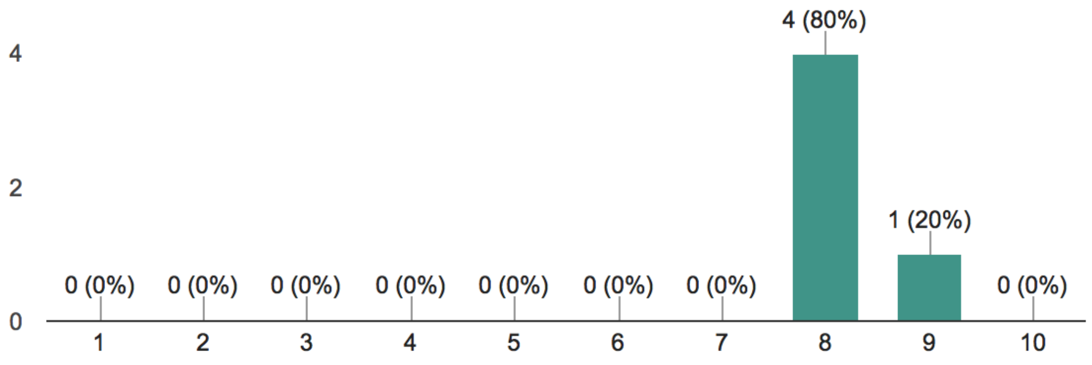


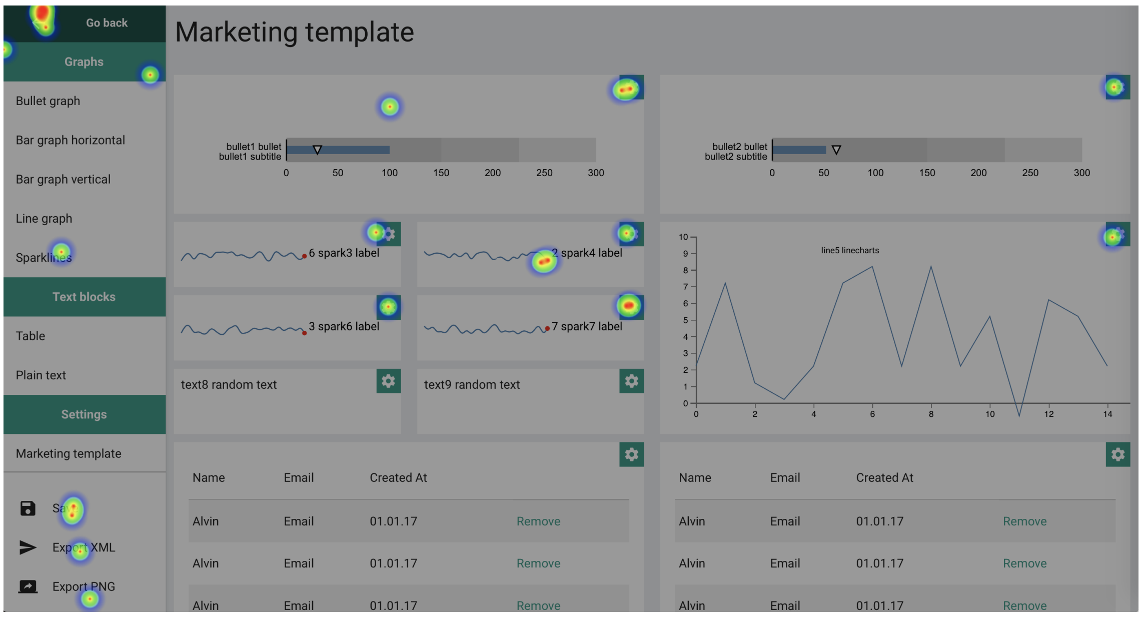
Figure 8.7: Usability evaluation results for general tasks related with dashboarts (editing, saving, exporting in different formats).

Obtained results show that usability of the application is quite high and that all primary tasks can be done without any efforts. Some of the users had difficulties with minor things, like finding account-related parts of the application. To make it easier, Configuration menu tab was renamed to Account.

While performing usability evaluation, it is important to keep in mind users context [1]. When the user knows that somebody is watching him, or that he may be recorded, his behavior may change. To avoid this, Google Form[2] was created and sent to all participants, asking to take a test in some calm environment.

To get the idea of users behavior on every page, Yandex Metrika Webwisor[3] tool was used. It creates different kinds of reports about website visits, including: clicks heatmaps, link maps, scrolling maps, recording of a user's screen during the visit, etc. For needs of this evaluation, clicks heatmaps were the most important. From them it is visible if a lot of users tried to make some actions which were not specified in the questionnaire. If this were the case, that would mean that the tasks are not obvious, or particular functionality is hidden in the interface. And from videos created with the help of Webvisor it is possible to understand the exact order of website navigation. These videos may also be used to estimate a time which was needed for a user to perform each task. Of course, such estimations are not exact ones, but they are still precise enough to compare the time needed to finish different sections.

The first analyzed heatmap is the page for editing a dashboard. It is selected since on this page it is possible to do almost all tasks, which were specified as the most important ones for the primary persona. From Figure 8.8 it is visible that main functionality is easily discoverable, since all clicks are gathered in correct places, and there weren't a lot of clicks on application parts without any functionality.



Name	Email	Created At	
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove

Name	Email	Created At	
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove

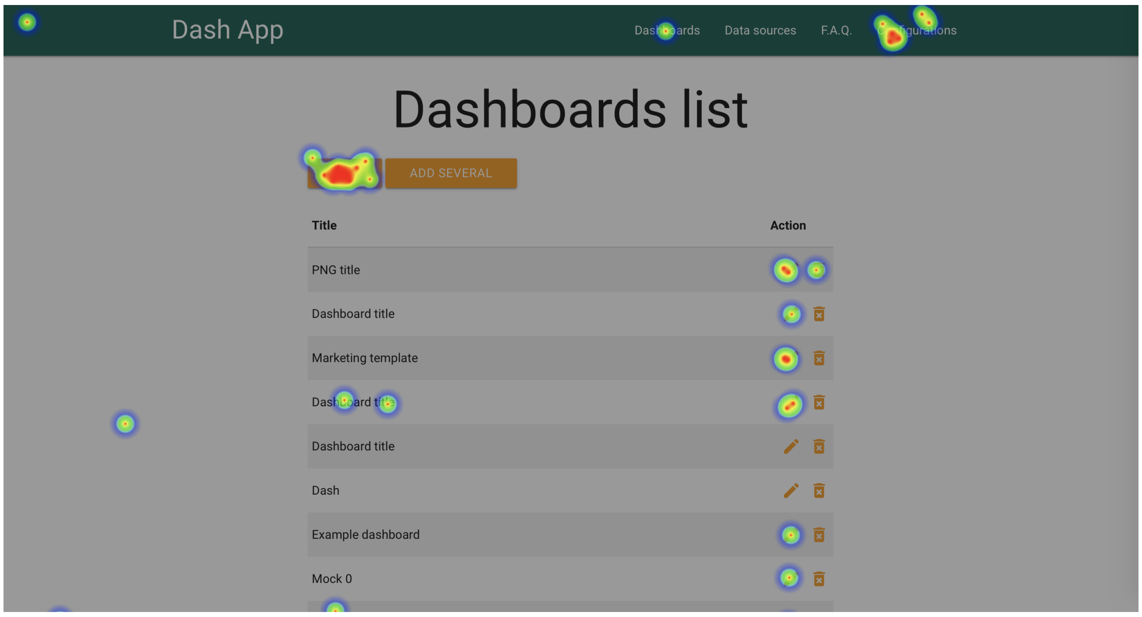
Figure 8.8: Heatmap generated for Marketing template dashboard.

Heatmap for a page with a list of all dashboards (Figure 8.9) though shows that to edit a dashboard some users tried to click on its title in a table. Indeed, icons are less usable

[2] https://goo.gl/forms/fn5oPGyKHIxVygBp2
[3] https://metrica.yandex.com/promo/webvisor/

elements, and more efforts are needed to discover them. So, to improve this dashboards title are made clickable too.

Full listing of a questionnaire together with the rest of the generated heatmaps may be found in the Appendix C.



Title	Action
PNG title	
Dashboard title	
Marketing template	
Dashboard title	
Dashboard title	
Dash	
Example dashboard	
Mock 0	

Figure 8.9: Heatmap generated for the index page.

Chapter 9

Conclusion

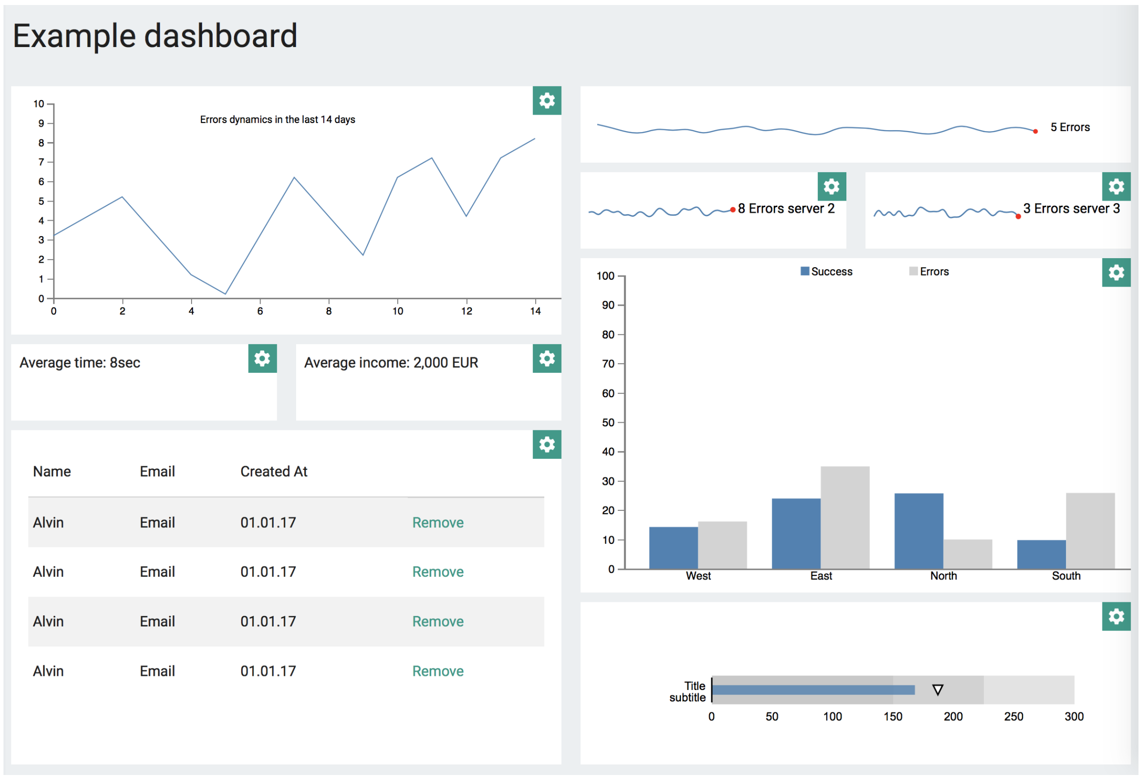
This paper describes a development process of a new application for the dashboards creation. Firstly, it was shown that there is a need for such a tool since none of the existing ones satisfy all requirements. Among main requirements are: make dashboards easily readable, follow UX practices based on how human brain works, allow creating dashboards quickly and with minimum input needed, create custom XML structure for the export and allow PNG export in order to achieve better results in the dashboard usability analysis. Based on all requirements which were developed throughout the paper application prototype was designed. To achieve described design quality, new JavaScript library – UXgraph was developed, containing reusable components for building the graphs. At its current stage, it is a Vue.js extension which uses D3.js for data visualization. The library was published as an *npm module*, and though it has only several types of graphs, and functionality is limited, it already had more than 900 downloads. This shows high demand on such a library.

The next phase described in the thesis was development and deployment of specified web application. It is developed using RESTful API as a backend for handling dashboards operations, so that client and server sides may be changed separately when needed, without influencing each other. Front-end part covers all tasks specified for the primary persona. Dashboards created with the developed application satisfy all ergonomic principles described in this thesis (Figure 9.1). The process of a dashboard creation is fast and easy.

Both back-and and front-end parts were deployed to DigitalOcean server[1][2]. To ensure that it satisfies all requirements, user testing and evaluation were performed. For this, five users from different target groups, including the target group of a primary persona, were asked to take part in a survey. The survey covered questions about the general application functionality. After the results of the testing were gathered and analyzed, several changed have been made to the application interface, in order to improve the interaction process. To increase application usability, it has an F.A.Q. section, covering primary interaction processes, comments in the code, and also a README file in its GitHub repository. So does UXgraph library as well.[3]

In the future, the application may be extended to allow more functionality (such as auto-generating of dashboards for testing purposes based on some generic algorithm), to support more graphs types or be converted to a hybrid mobile application (using Cordova

[1]Deployed application public IP: http://46.101.183.148/
[2]Deployed application domain name: http://uxdash.ml
[3]UXgraph source files: https://github.com/lirael/vuejs-d3-uxgraph



Average time: 8sec

Average income: 2,000 EUR

Name	Email	Created At	
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove

Figure 9.1: Example of a dashboard created with the developed application.

wrapper, for example). The application was developed keeping these possible extensions in mind, so as a result, it is easily scalable and well documented.

Bibliography

[1] Benyon, D.: *Designing Interactive Systems: People, Activities, Contexts, Technologies*. Addison Wesley. 2005. ISBN 0321116291.

[2] Cairo, A.: *The Truthful Art: Data, Charts, and Maps for Communication*. New Riders. 2016.

[3] Camões, J.: *Data at Work*. New Riders. 2016. ISBN 0134268636.

[4] Cleveland, W. S.; McGill, R.: *Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods*. Journal of the American Statistical Association. 1984.

[5] Cooper: *Sams - Pearson Education*. Indianapolis. 2004. ISBN 0672326140.

[6] Eckerson, W. W.: *Performance Dashboards: Measuring, Monitoring, and Managing Your Business*. Wiley. 2010. ISBN 0470589833.

[7] Few, S.: *Dashboard Confusion*. Intelligent Enterprise. 2004.

[8] Few, S.: *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly Media. 2006. ISBN 0596100167.

[9] Few, S.: *Data Art vs. Data Visualization: Why Does a Distinction Matter?* A blog by Stephen Few. May 2012. [Online; visited 12.12.2016].
Retrieved from: http://www.perceptualedge.com/blog/?p=1245

[10] Gartner: *IT Glossary*. [Online; visited 10.12.2016].
Retrieved from: http://www.gartner.com/it-glossary/

[11] Healy, P. M.; Palepu, K. G.: *The Fall of Enron*. Journal of Economic Perspectives. 2003.

[12] Holmes, S.: *Getting MEAN with Mongo, Express, Angular, and Node*. New Riders. 2010. ISBN 0321683684.

[13] Iliinsky, N.: *Four pillars of visualization*. A presentation by Noah Iliinsky. [Online; visited 10.12.2016].
Retrieved from: http://www.slideshare.net/CindyXiao/four-pillars-of-visualization-by-noah-iliinsky

[14] Johnson, J.: *Designing with the Mind in Mind, Second Edition: Simple Guide to Understanding User Interface Design Guidelines*. Morgan Kaufmann. 2014. ISBN 0124079148.

[15] Moere, A. V.; Purchase, H.: *On the role of design in information visualization.* Information Visualization. 2011.

[16] Pastushenko, O.: *UXgraph - Vue.js library with predefined D3 graphs.* [Online; visited 10.05.2017].
Retrieved from: http://excel.fit.vutbr.cz/submissions/2017/059/59.pdf

[17] Rosenberg, D.; Grafton, A.: *Cartographies of Time: A History of the Timeline.* Princeton Architectural Press. 2012. ISBN 1616890584.

[18] Tufte, E. R.: *The Visual Display of Quantitative Information.* Graphics Press. 2001. ISBN 1616890584.

[19] Tufte, E. R.: *Beautiful Evidence.* Graphics Press. 2006. ISBN 1930824165.

Appendices

List of Appendices

Appendix A

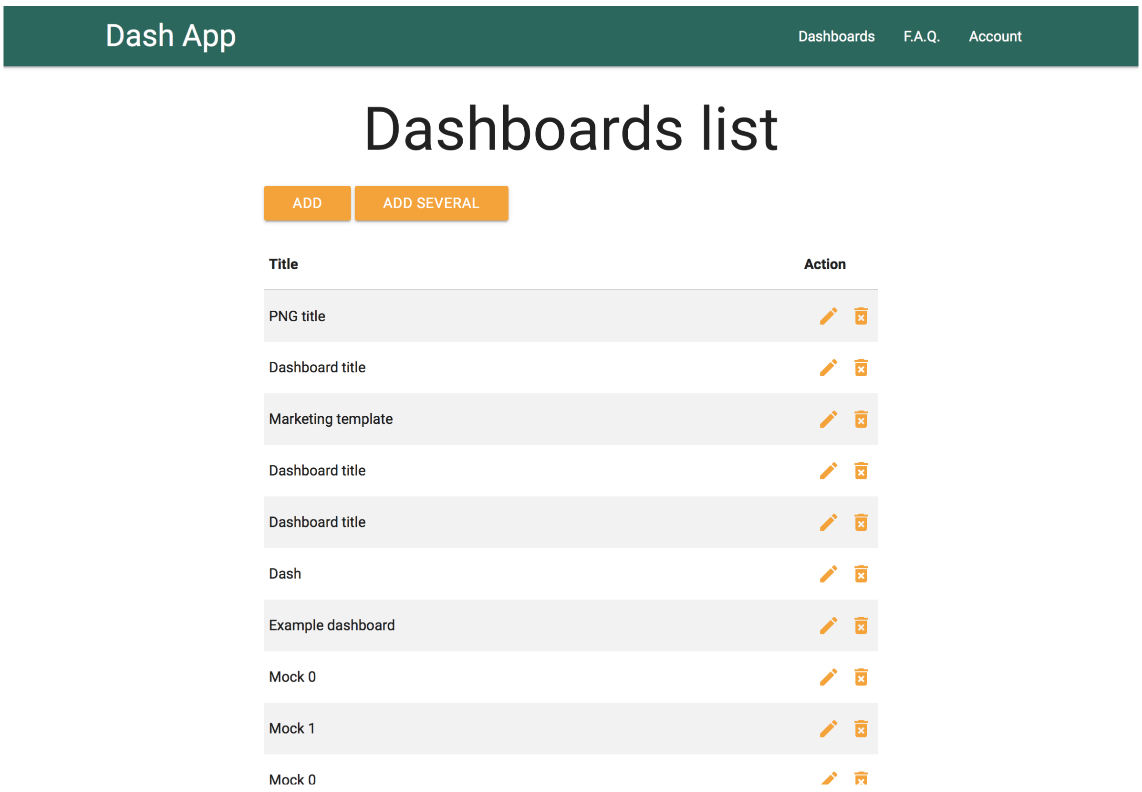
Attached CD content

On the attached CD there are following files and folders:

```
\theory
  \src
    <LaTeX source code>
  xpastu01.pdf // Thesis text in PDF format
\dashboard-api
  <backend-end source code>
\dashboard-app
  <front-end source code>
  \build
    <front-end build files>
README // Technical information about the application, deployment process
content.txt // CD content explanation
manual.pdf // User manual from the interaction with the deployed application
```

Appendix B

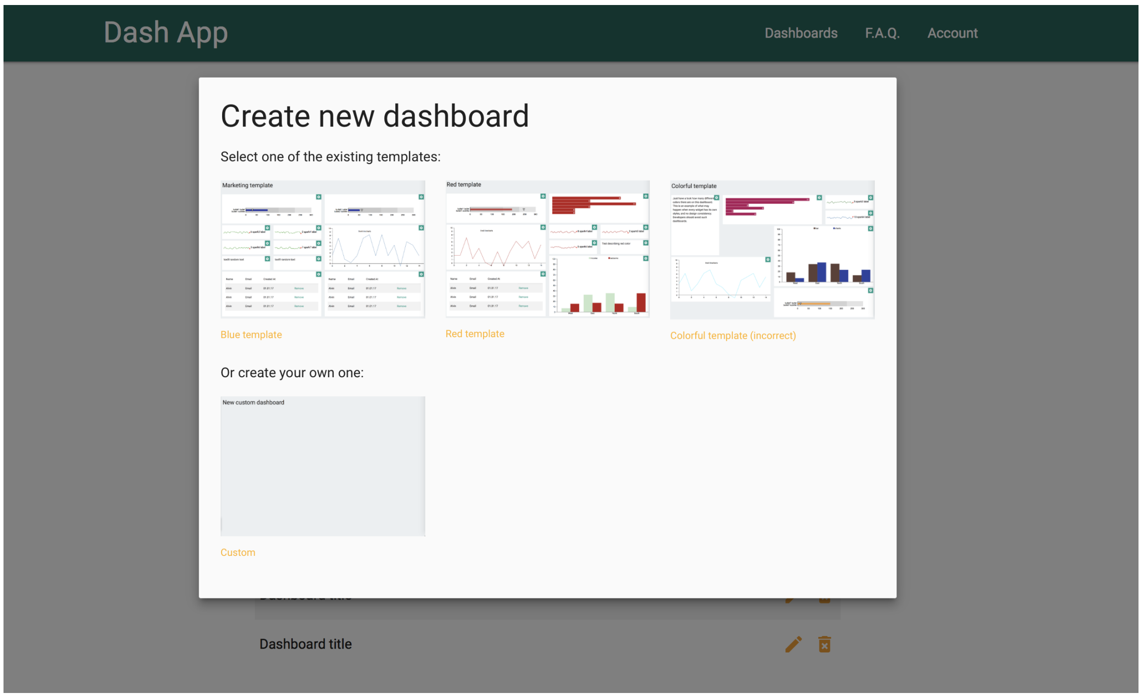
Application screenshots



Dashboards list

Title	Action
PNG title	
Dashboard title	
Marketing template	
Dashboard title	
Dashboard title	
Dash	
Example dashboard	
Mock 0	
Mock 1	
Mock 0	

Figure B.1: Page listing all dashboards



Dash App

Create new dashboard

Select one of the existing templates:

Blue template

Red template

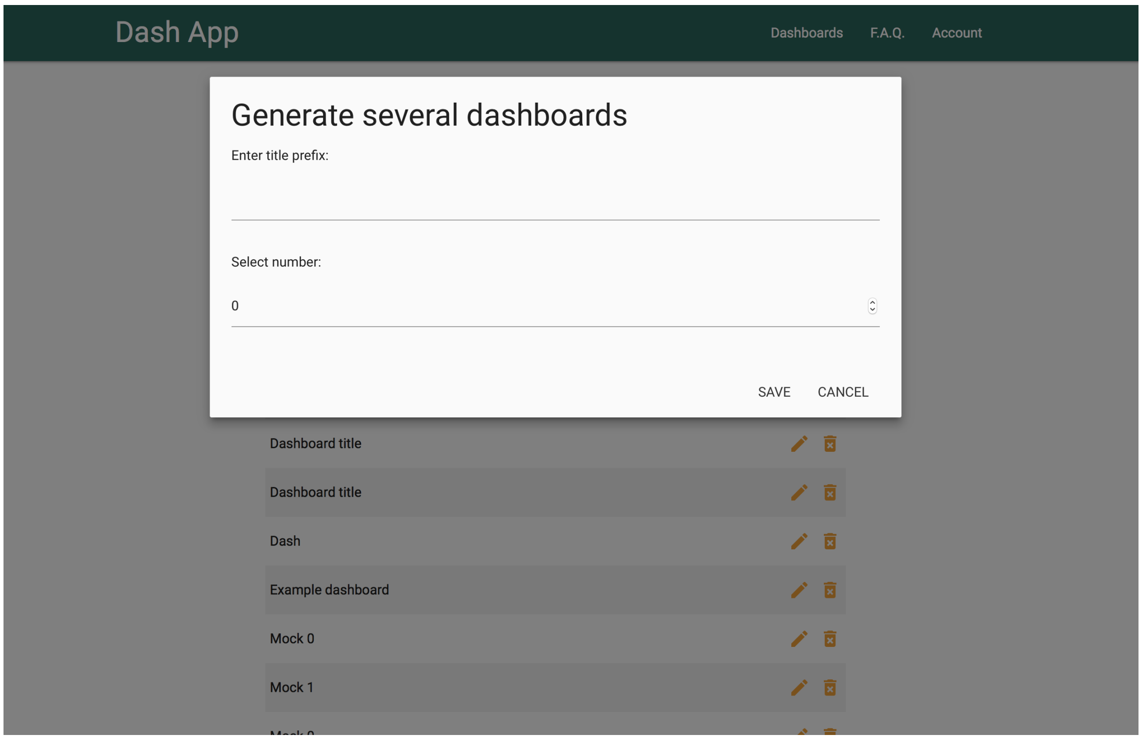
Colorful template (incorrect)

Or create your own one:

Custom

Dashboard title

Figure B.2: Create new dashboard form



Dash App

Generate several dashboards

Enter title prefix:

Select number:

0

SAVE CANCEL

Dashboard title

Dashboard title

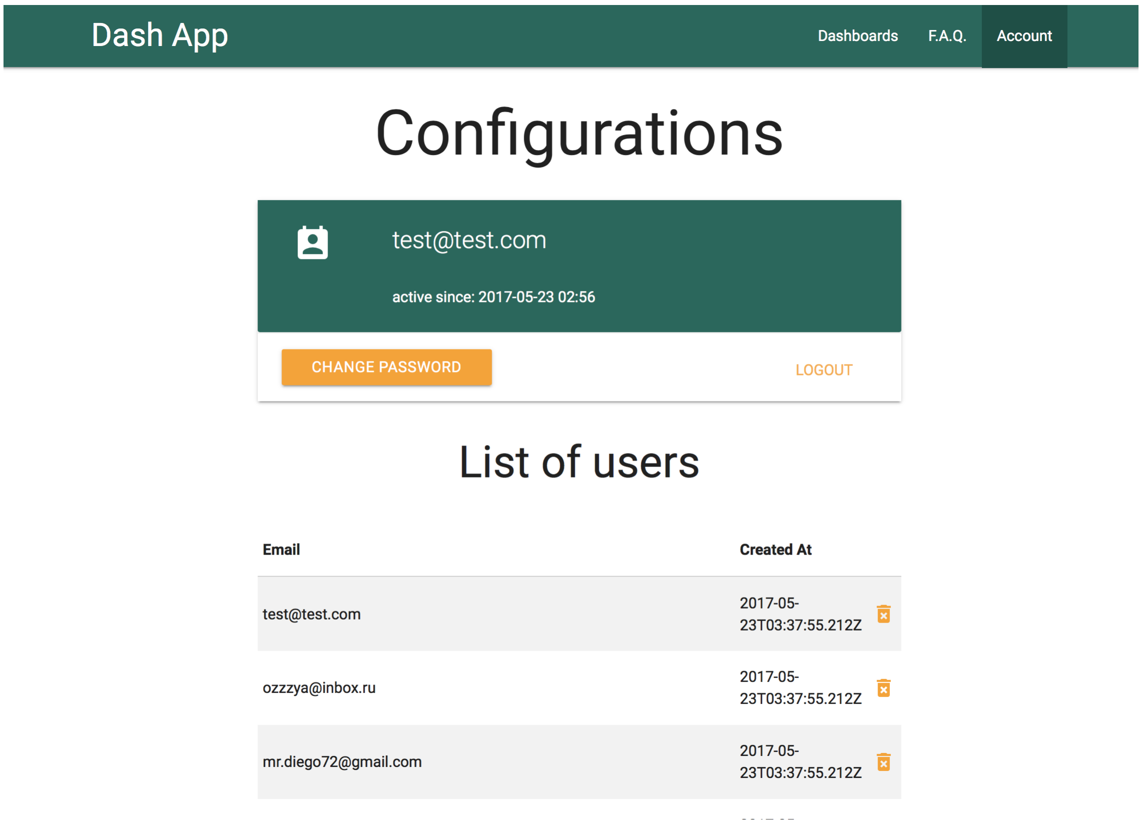
Dash

Example dashboard

Mock 0

Mock 1

Figure B.3: Generate several dashboards form



Dash App

Dashboards F.A.Q. Account

Configurations

test@test.com

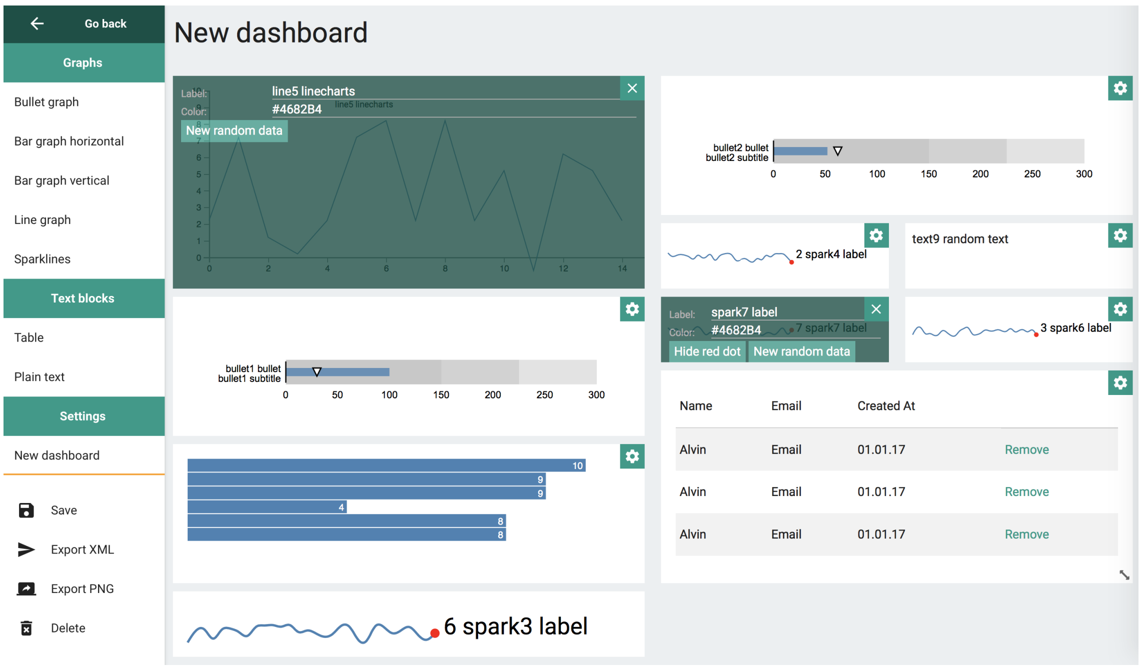
active since: 2017-05-23 02:56

CHANGE PASSWORD LOGOUT

List of users

Email	Created At
test@test.com	2017-05-23T03:37:55.212Z
ozzzya@inbox.ru	2017-05-23T03:37:55.212Z
mr.diego72@gmail.com	2017-05-23T03:37:55.212Z

Figure B.4: Account configurations page



Go back

Graphs

Bullet graph

Bar graph horizontal

Bar graph vertical

Line graph

Sparklines

Text blocks

Table

Plain text

Settings

New dashboard

Save

Export XML

Export PNG

Delete

New dashboard

Label: line5 linecharts

Color: #4682B4

New random data

text9 random text

Label: spark7 label

Color: #4682B4

Hide red dot New random data

Name	Email	Created At	
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove
Alvin	Email	01.01.17	Remove

Figure B.5: Edit a dahboard page

Appendix C

Summative usability evaluation materials

Listing of the questionnaire:

```
This usability testing is aimed to evaluate the usability of a software
for working with dashboards, developed as my Thesis project.
Please, follow instructions below and write down all your suggestions
and difficulties during the interaction with the app in the text field before.
Every your idea or emotion is valuable! :)
All evaluation shouldn't take more then 10 minutes of your time!
Thanks

Section 1

1. Your age
2. Your sex
3. Your occupation

Section 2

1. Go to url: http://46.101.183.148
2. Create new account
3. Log in to the application with your new account
4. Find out how to see your profile and change a password
5. Find and have a look at the instructions about the software usage
6. How easy and pleasant was this part?

Section 3

1. Create a new dashboard with a custom template
2. Edit the title of a dashboard
3. Try adding new widgets of different types
4. Delete a widget from a dashboard
5. Add a sparkline widget to the dashboard
6. Change a color on a sparkline widget you've just added
```

7. Change size of a sparkline widget you've just added
8. Save a dashboard
9. How easy and pleasant was this part?

Section 4

1. Try to export a dashboard as an XML
2. Try to export a dashboard as a PNG
3. Go to the list of all dashboards. Try to edit another dashboard
4. Delete any dashboard
5. Create a new dashboard using some template
6. How easy and pleasant was this part?

Section 5

Thanks!

1. Let me know if you have some general comments about the application

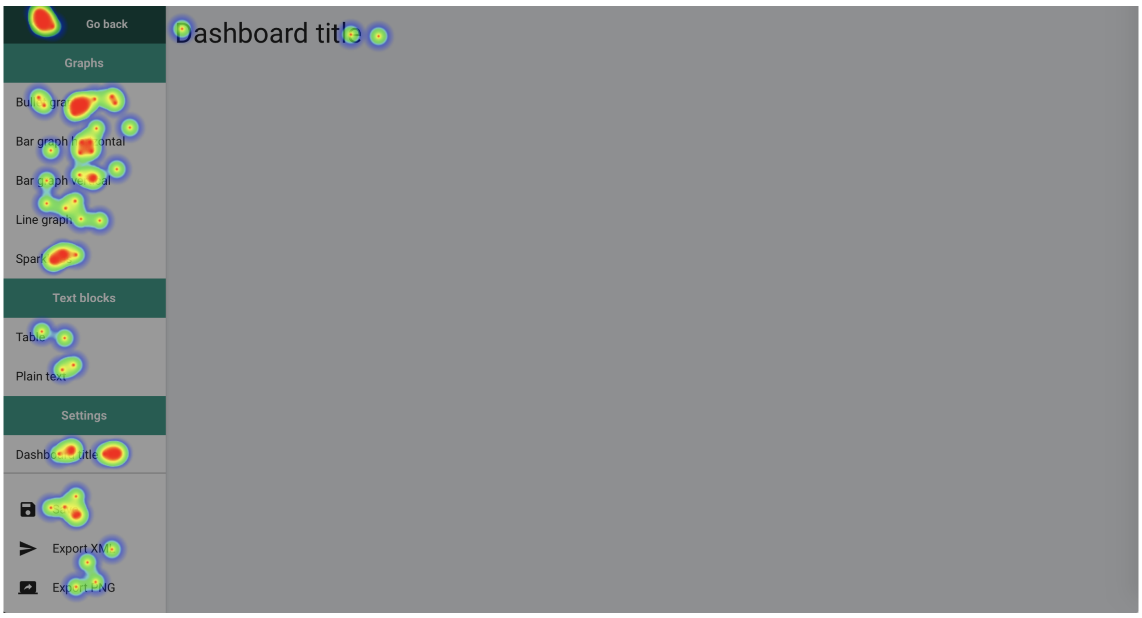


Figure C.1: One of the resulting heatmaps