TU WIEN Informatics

Development and Evaluation of an Online Style Guide

MineralBay as a Case Study for Development and Evaluation of an Online Style Guide

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Software Engineering and Internet Computing

eingereicht von

Hafize Okunakol
Matrikelnummer 0826496

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Thomas Grechenig
Mitwirkung: Christoph Wimmer

Wien, 9. Oktober 2020

Unterschrift Verfasserin

Unterschrift Betreuer

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Development and Evaluation of an Online Style Guide

MineralBay as a Case Study for Development and Evaluation of an Online Style Guide

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Software Engineering and Internet Computing

by

Hafize Okunakol
Registration Number 0826496

to the Faculty of Informatics

at the TU Wien

Advisor: Thomas Grechenig
Assistance: Christoph Wimmer

Vienna, 9th October, 2020

Signature Author

Signature Advisor

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Development and Evaluation of an Online Style Guide

MineralBay as a Case Study for Development and Evaluation of an Online Style Guide

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Software Engineering and Internet Computing

eingereicht von

Hafize Okunakol
Matrikelnummer 0826496

ausgeführt am
Institut für Information Systems Engineering
Forschungsbereich Business Informatics
Forschungsgruppe Industrielle Software
der Fakultät für Informatik der Technischen Universität Wien

Betreuung: Thomas Grechenig

Wien, 9. Oktober 2020

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Hafize Okunakol

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 9. Oktober 2020

Hafize Okunakol

Kurzfassung

MineralBay ist ein webbasiertes Materialverwaltungssoftwaresystem für Aushubmaterialen und befindet sich derzeit in Entwicklung. Das Design der Applikation wirkt aufgrund fehlender Richtlinien und Verantwortlichen heterogen. Die Entwickler haben nicht die notwendige Zeit und das Know-how über Usability um Designentscheidungen zu treffen. Die folgende Arbeit zeigt einen Lösungsansatz und gibt den Entwicklern einen Styleguide als Instrument zur Entscheidungsfindung und Informationsbereitstellung.

In der Diplomarbeit wurde ein Usability-Konzept für die Webapplikation MineralBay ausgearbeitet. Die Ergebnisse der durchgeführten Usability-Recherche wurden den Entwicklern als ein Online-Styleguide bereitgestellt. Der Fokus der Forschung lag auf der Integration des Styleguides in den Software-Entwicklungsprozess und der Überprüfung der Akzeptanz eines Styleguides und der Erwartungen an dessen Inhalt.

Grundlagen für das Konzept der neugestalteten Applikation und des implementierten Styleguides ergeben sich aus einer ausführlichen Literaturrecherche zum Thema Usability und Styleguides. Nach einer theoretischen Aufbereitung wurde im Rahmen einer Nutzeranalyse eine quantitative Erhebung in Form eines Online Fragebogens mit den potentiellen Benutzern und den Entwicklern von MineralBay durchgeführt. Diese Analysen resultierten in Personas und den Designentwürfen in Form von Wireframes. Die Wireframes dienten als Input für den Inhalt des Styleguides. Um den Styleguide zu evaluieren und eine Aussage darüber treffen zu können, ob die Anforderungen der Entwickler erfüllt wurden, wurden kontextuelle Analysen durchgeführt.

Die gewonnenen Erkenntnisse zeigen, dass sich Styleguides mit dem Styleguide getriebenen Ansatz von De La Cuadra [DLC15] in den Entwicklungsprozess eines Softwareengineering-Projektes integrieren lassen. Die hohe Akzeptanz eines Styleguides seitens der Entwickler basiert auf den Erwartungen an Konsistenz, einfachere Entscheidungsfindung und die daraus resultierende verkürzte Entwicklungszeit. Die aus der Literatur erarbeiteten Empfehlungen zum Inhalt eines Styleguides sind grundsätzlich ausreichend, damit der Styleguide bei Entwicklern Verwendung findet. Darüber hinaus sehen die Entwickler einen Bedarf an zusätzlichen technischen Informationen zur Einbindung der Komponenten in den aktuellen Entwicklungskontext sowie einem Stylesheet zum Downloaden, das die CSS-Codes aller Komponenten zusammenfasst.

Keywords: *Styleguide, Usability, Webapplikation, User Centered Design, Human Computer Interaction*

Abstract

MineralBay is a web-based material management software for excavated material and is currently under development. The design of the application seems rather heterogeneous due to missing guidelines and responsible persons. The developers do not have the necessary time and the usability know-how to make design decisions. The following work demonstrates a solution approach and gives developers a style guide as a decision making and information sharing tool.

In this thesis, a usability concept for the web application MineralBay has been developed. The results of the usability research were provided to the developers as an online style guide. The focus of the research was on the integration of the style guide in the software development process of university projects, the review of the acceptance of it and the expectations of the developers towards the content.

The basics for the concept of the redesigned application and the implemented style guide are the result of a detailed literature research on the topic of usability and style guides. After a theoretical treatment, a quantitative survey in the form of an online questionnaire with potential users and the developers of the MineralBay was carried out. This analysis resulted in personas and design drafts in the form of wireframes. The wireframes served as an input for the content of the style guide. To evaluate the style guide and be able to make a statement about whether the requirements of the developers have been met, contextual analyzes were performed.

The findings show that style guides can be integrated into the development process of a software engineering project using the style guide driven development concept by De La Cuadra [DLC15]. The high acceptance of a style guide by developers is based on expectations of consistency, easier decision-making and the resulting reduced development time. The recommendations from the literature on the content of a style guide are basically sufficient for the style guide to be used by developers. In addition, the developers see a need for additional technical information to integrate the components into the current development context as well as a downloadable stylesheet that summarizes the CSS codes of all components.

Keywords: *style guide, usability, web application, user centered design, human computer interaction*

Contents

CHAPTER 1

Introduction

1.1 Problem Description

The enormous increase in individual mobility and the focus on urbane habitats increasingly relocate roads to the underground. In the course of tunnel construction projects, up to 800 million tons of tunnel excavation material will arise in the near future. Currently, most of this excavation material is stored in landfills. A research project of the Montan University called MineralBay has the goal to implement a web application, which contains specific information about future European tunnelling and underground engineering projects, such as geological and mineralogical parameters, the construction time, the distance to existing storage facilities and the processing industry, etc. The web application will provide search and input masks to enter resource information found during the construction of tunnels. Style guides are an integral part in today's web development to guarantee the consistency of a web application and thereby convey a picture of unity. The creation of the style guide and its integration into the software development process represents a challenge that has a decisive impact on the acceptance of the style guide.

1.2 Motivation

Nowadays, increasingly more information is produced and provided using the web. The web as a tool provides not only a variety of websites for finding information, also management tasks can be done online. More and more software applications are built using web technologies [TM11]. They are commonly referred to as web applications. Such a web application can convince the user, that tasks can be accomplished quickly. The fact that many people with different previous knowledge, education and experience can use a web application requires a uniform usability concept. Modern system developments include

users or user groups at an early stage in the development and design processes. The aim of such practiced usability engineering is to already support a task- and user-appropriate functionality and operability during the formation [Hol05].

A usability concept for the web application MineralBay needs to be worked out, so that the interaction is best adapted to the sensory and cognitive abilities and limitations of human beings. Usability is a measure for how effective, efficient and satisfactory, the user can achieve his goals with the product [Nie03, Hol05]. Hence, usability is of great importance for web applications and has to be planned properly. It is important for the success of a product that its appearance and corporate identity are consistent. With increasing size of the product, the management of the corporate identity becomes more difficult [Deb]. Style guides are an important tool in ensuring the consistency of a product and providing a sense of unity. In addition to standardizing the appearance of an application, a style guide offers other benefits as well. The reusability of components and structures is made possible, which in turn facilitates the further development and maintenance. In addition, a style guide can be used as a communication tool within the development team and with stakeholders. A style guide driven development approach puts such a style guide in the foreground of the development and sees it as a driver. It separates the development of the frontend from the development of the backend. Design-relevant changes are only implemented in the style guide and made available to the project team for applying to the user interface of the web application.

1.3 Expected Results

Central terms for the scientific embedding of this topic are Human-Computer Interaction (HCI) [DFAB04, OB08] and web usability [McK01, MRC06]. The focus of this work lies primarily in the field of web usability.

The thesis is concerned with the appropriate application of web usability and human computer interaction concepts to determine a usability concept that is best suited for MineralBay. The application is to be redesigned and a uniform look & feel is to be introduced. This goal makes the systematic design process challenging and complex, especially since it should take into account many factors that can not be quantified in advance. In a first step, fundamental principles, recommended practices and patterns that are relevant for usability design will be identified and defined. The elaborated theoretical work will serve as a foundation for the realization of the graphical interface of the web application MineralBay. The user interface elements as well as the programming code behind them, the design decisions and guidelines, which ensure a uniform design, will be made accessible in an online style guide.

There are many case studies carried out in software engineering, but there are no specific text books on how to perform case study research in this domain [HR07]. This case study

research combines and integrates qualitative and quantitative research methods, which is referred in the literature as "mixed-method" [MNL+93, Fli08] and uses checklists as a guide [RH09, WCFD+12, Wie10]. According to Flick [Fli08] triangulation is important to increase the precision of empirical research. This is a mixed-method approach and aims to consider a research question from multiple perspectives. For this purpose the data is collected or evaluated with different methods. The following data collection and evaluation methods with focus on different user groups will be accomplished [PB13]:

- Questionnaire-based survey with potential users of MineralBay
- Questionnaire-based survey with the developers of MineralBay
- Interviews and contextual inquiries with the developers of MineralBay

Based on the empirical findings, the reusable design artifacts, the online style guide will be improved.

The research questions to be answered by this thesis are as follows.
Research Question 1:
There are many approaches in the literature to integrate a style guide to the software development process. Is the style guide driven development approach suitable for applying a new design to an existing web application?

Research Question 2:
What content must be provided, so that a style guide is suitable for developers?

1.4 Structure of the Thesis

This thesis is structured as follows:

- **Introduction:** This chapter gives a short introduction into the motivation, the goals and the related research questions of the current work.
- **Fundamentals of Web Usability:** This chapter covers the fundamentals of web usability with a focus on the current state of the literature regarding style guides.
- **Methodical Procedure - Case Study:** In this chapter the context of the case study is discussed and a brief overview of the methodical procedure of the thesis is given.
- **User and Requirements Analysis:** The findings of the user and requirements analysis conducted with the developers and potential users of MineralBay are given in this chapter.

- **Implementation:** This chapter discusses the wireframes for MineralBay and the implementation of the style guide. An overview of the used architecture, the implementation itself and the used technologies is given.
- **Evaluation and Results of the Contextual Inquiry:** The results of the evaluation of the style guide are presented in this chapter.
- **Discussion:** The chapter discusses the findings and the process of the current work and answers the research questions.
- **Conclusion and Outlook:** This chapter summarizes the key findings and gives an outlook to possible next steps in the future.

CHAPTER 2

Fundamentals of Web Usability

This master thesis deals with the development and evaluation of an online style guide using the case study of the tunnel project management system MineralBay. However, to understand what style guides are and what their purpose is, the theoretical context they are embedded in must be explained in more detail. This chapter describes the basic concepts and foundations, which are necessary for an understanding of the problems and usability issues mentioned in this thesis. First, a general introduction to the subject area is given, which make up the theoretical background of the research. Afterwards an overview of existing solutions to the problem or related problems are shown.

2.1 Human-Computer Interaction

HCI, is an interdisciplinary research field that focuses on design, implementation and evaluation of interactive technologies. It is characterized by interdisciplinary cooperation of psychology, cognitive science and pedagogy, linguistics and semiotics, sociology and anthropology, art and design, ergonomics, computer science and artificial intelligence, as well as engineering sciences [Car13]. Figure 2.1 illustrates the disciplines involved in contemporary design of HCI. Initially it emerged as a specialty area in computer science embracing cognitive science and human factors engineering and was referred as Computer-Human Interaction (CHI).

Special Interest Group on Computer Human Interaction of the Association for Computing Machinery (ACM SIGCHI) defines HCI as follows: „Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.“

Winograd et al. [WF87] characterize HCI by three factors:

Figure 2.1: Variety of disciplinary knowledge and skills involved in contemporary design of HCI ([Car13])

- the man with his anatomical, physiological and psychological characteristics, communication intentions and his actions
- the task to edit or resolve with the help of the computer
- the technology that has to create the best possible conditions to solve the tasks effectively, efficiently and satisfactorily

The aim of the optimal interaction of the three factors and hence the optimized interaction between man and computer is to increase the effectiveness of the interaction of human and technical services. Here psychological and psycho-physical stress optimization is as essential as the stabilization of the mental health and personality development. Also the optimization of the learning opportunities plays an essential role. If this target is at least partially missed, interaction disorders in form of task, function and action problems appear [Sha85]. As a result, users will need more time to learn and handle the software. They make mistakes, feel dissatisfied, use it involuntarily, may never learn to use the full functional scope of the software and they will not potentially buy it.

Preece et al. [PRS+94] introduced several factors, HCI specialists have to consider in order to achieve a safe and user-friendly system. Many of these factors interact with each other, making the design even more complex. (see Fig. 2.2)

A superordinate domain is the Human-Machine Interaction, which deals with similar issues, but generalizes the interaction partner of the man to machine.

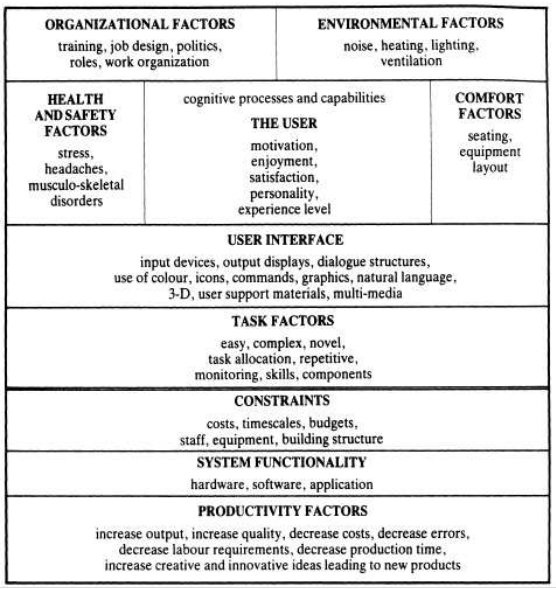


Figure 2.2: Factors for the design of HCI ([Car13])

2.2 Usability

Usability (also referred as Web Usability in context of a website or web application) is located in the scientific field of Human Computer Interaction (HCI) and deals with the research of communication between humans and computers [RF07]. The term was coined in the early 80s, to replace the inexact concept of "ease of use". In contrast to "ease of use", usability is not only limited to a comfortable use, but also takes into account the appropriate support of the user to achieve its objectives in the specific application context [SH06]. As in all other informations systems with a user interface, improving the quality of the offered service and thus the satisfaction of the user form the basics for the usability of web sites (also referred as Web Usability).

Since a variety of definitions and explanations are available, it is difficult to find a universal definition [HH93, PRS+94, Sha85, Shn97]. The most commonly used definition is by the International Organization for Standardization (ISO). According to ISO 9241-11 [ISOa] usability of a product is defined as the extent to which it can be used by a specified user to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. In the following, these terms are discussed in general and additionally in the context of information retrieval and knowledge acquisition.

Effectiveness describes how exactly and completely a particular product can fulfill requirements or tasks. Closely connected with these is the design of the menu, navigation

and orientation elements, which ensure the fulfillment of the tasks. Effectiveness as well efficiency determines how much cognitive effort a user needs to handle the system and how many cognitive resources for the actual acquisition of knowledge remain available afterwards. The extent to which the system helps the user to achieve its goals in full and in high quality, can be derived by whether the objective has been achieved at all, which and how much relevant information has been accessed and which and how much potentially relevant information has not been used.

Efficiency indicates that the effort used should be as low as possible in order to achieve a certain goal. The acquisition of knowledge is perceived as efficiently when the resources used as the effort to achieve the goals or maintaining the motivation commensurate with the results achieved. Efficiency can only be found to be satisfactory if few clicks and keystrokes are necessary to reach and work through the appropriate knowledge content. The degree of efficiency with which an application can be handled is often determined over time. The efficiency corresponds to the percentage, how quickly and well knowledge is achieved. However, it is important to note that the efficiency is at least dependent on the error rate of the system and may also affect the criterion satisfaction.

Satisfaction is a measure of positive attitude of the user towards the use of the product as well as its freedom from impairment by the product [Dah06]. In addition to the requirements associated primarily with reaching the goal by a person, it is also important to consider the subjective perception of this person, which is described with the satisfaction of the user. Satisfaction is an aspect of usability, which influences all other aspects. It reflects the acceptance of the system by the user and answers the question of whether the functionality is regarded as appropriate for the acquisition of knowledge and whether it sufficiently supports achieving goals. Satisfaction determines to a significant extent whether the user is ready to use the offer. There are several possibilities to determine the satisfaction. It can be performed very complex by measuring the eye movements and by evaluating pupil dilations. A less expensive way to determine satisfaction is conducting surveys. Moreover, spontaneous expressions of the user during the use of the system can be logged and the frequency of positive and negative comments can be documented. These can be expressions of despair or frustration, but also of amazement or surprise. It is essential to clarify the occurence of these expressions in an interview with the user afterwards.

According to Nielsen[Nie12] usability is not a single characteristic of a product and not measured as such; it is part of the utility of a product and is a measurement of quality that user is experiencing when interacting with a system. Nielsen identifies five distinct attributes of usability: efficiency, satisfaction, learnability, memorability and errors. ISO agrees on efficiency and satisfaction. As stated by Nielsen beside efficiency and satisfaction good usability is formed of:

Learnability allows the user to quickly display knowledge and skills using the system. Once the user has learned some basic functions, such as handling the navigation the user

must be able to start getting work done with the system. A measure of learnability is the time it takes a user to achieve a certain level of skills in using the system. Very long training periods indicate weaknesses of learnability, so that a revision of the usability is needed.

Memorability means that the systems should be easy to remember, so that the user is able to return to the system after some period of not having used it. There are two ways to measure memorability. The first way measures times for processing certain tasks on the systems with two different users. The one user is an occasional user and the other one works frequently with the system. Both times are compared. Large differences indicate usability flaws. The second method carries out memory tests only with users who work occasionally with the system. The users have to remember, which effects certain commands have.

Errors mean a low error rate and the help offered to the user in handling or correcting errors quickly and easily. In this case errors are actions, which do not lead to the desired result. Measuring the error rate is easy since they only need to be summed up. It is more difficult to determine whether assistance offered also has been understood and applied to resolve the errors.

In addition to the attributes discussed above, there are other attributes that describe usability. The following table shows different views on the attributes of usability:

Authors	Attributes
Abran et al. (2003) [AKSS03]	effectiveness, efficiency, learnability, satisfaction, security
Booth (1989) [Boo89]	usefulness, effectiveness, learnability, attitude
Brinck et al. (2002) [BGW02]	functionally correct, efficient to use, easy to learn, easy to remember, error tolerant, subjectively pleasing
Clairmont et al. (2000) [DM00]	successfully learn and use a product to achieve a goal
Dumas et al. (1999) [DR99]	perform tasks quickly and easily
Furtado et al. (2003) [Fa03]	ease of use and learning
Gluck (1997) [Glu97]	useableness, usefulness
Guillemette (1995) [Gui95]	effectively used by target users to perform tasks
Hix et al. (1993)[Glu97]	initial performance, long-term performance, learnability, retainability, advanced feature usage, first impression, long-term user satisfaction
ISO (1998)	effectiveness, efficiency, satisfaction
Kengeri et al. (2014) [KSH+14]	effectiveness, likeability, learnability, usefulness
Kim (2002) [Kim02]	interface effectiveness
Nielsen (1993)	efficiency, learnability, memorability, errors, satisfaction
Oulanov et al. (2002) [OP02]	affect, efficiency, control, helpfulness, adaptability
Preece et al. (1994)	learnability, flexibility, throughput, attitude
Seffah et al. (2006) [SDKP06]	effectiveness, efficiency, learnability, productivity, satisfaction, safety, trustfulness, accessibility, universality, usefulness
Shackel (1986, 2009) [Sha86, Sha09]	effectiveness, learnability, flexibility, user attitude
Quesenbery (2003)	effectiveness, efficiency, engaging, error tolerant, easy to learn

Table 2.1: Attributes of usability

Thomas [Tho98] divided the attributes of usability into three main categories, namely: outcome, process and task. The outcome category includes effectiveness, efficiency, and satisfaction. The process includes ease of use, interface, learnability, memorability, and error recovery. The task group includes functionality and compatibility.

Usability can also be grouped into two large categories:

- Inherent usability: is mainly related to the functional or dynamic part of interface usability. It includes attributes that focus on how to make the product easy to understand, easy to learn, efficient to use, less erroneous, and pleasurable [KK95].
- Apparent usability: is more related to the visual impression of the interface [KK95, Tra97].

Quesenbery [Que03] criticizes the definition of ISO 9241-11 in several points:

- It is too focused on well-defined tasks and goals, either ignoring the less tangible elements of user experience or forcing simplistic definitions of tasks.
- The emphasis on efficient and effective as the most important attributes of an interaction make it difficult to talk about how usability applies to products or context where these are less important. Work that looks at pleasure, engagement, or other difficult to measure emotional aspects are often defined as "beyond usability."
- Satisfaction is not a robust enough term to cover the needs in many situations.
- It may have been acceptable in a context of enterprise, but not in the consumer world of shopping, information-seeking and online services.

Quesenbery build on the definitions of Nielsen and ISO 9241-11 and developed five dimensions, which can be used in a Web site setting as well as for software development. These dimensions are effectiveness, efficiency, engagement, error tolerance, and ease of learning.

The different definitions and approaches trying to define usability are an indication of that there are no generally valid guidelines with which a high usability can be achieved. It is important to carefully analyze and evaluate the attributes linked with the application in order to develop the best possible design. The attributes of the definition above, lead to clarification of important aspects of development, such as: What is the profession of the users? How old are they? Will the application be used on a mobile device? For more precise answers, these and many other questions need to be answered at an early stage of a development cycle.

2.3 User Centered Design

Methods used in the design of the software application are mostly concerned with technical requirements. However, user requirements as well as the technical and functional requirements for the software are equally important. User-Centered Design (UCD) is a multidisciplinary design approach, which aims to reflect the user perspective to the available system design. It is based on knowledge and experience in various fields such as psychology, design and software development. In UCD the user is systematically included

and actively takes part in the design process, which makes it effective in overcoming the limitations of traditional system-centered design [MVSC05]. The main objective is to improve the usability and make easy to learn and easy to use interactive products that allow the user optimal interaction and a comfortable user experience.

The term has gained in popularity after the publication of the book "New Perspectives on Human-Computer Interaction" by Norman and Draper in 1986 [ND86]. This book draws the attention for the first time to the fact, that a system is designed for a user and so their needs should be at the foreground of the development.

According to Norman UCD "is simply a process that starts at the user's needs rather than the technology." The goal is a technology that assists the user. Thus, user-centered design requires developers who understand users.

Gould et al. [GL85] identified basic principles of a UCD process. These are as follows:

- Early focus on users and task: In order to understand the characteristics, attitudes, objectives, skills, needs and expectations of the target audience it is important to directly observe the users for example through interviews, surveys, direct participation in the work process.
- Early and continuous testing of the usability: Developers observe the users when using the system and get some feedback. Both, the users and the developers carve out improvements based on the observations and feedback.
- Iterative design: The usability problems arise gradually and corrections can cause new problems. For this reason the development goes through many cycles, in order to redesign the system according to the expectations of the user.
- Integrated design: All usability aspects should evolve in parallel. Otherwise, many solutions would not be an advantage for users, because they are in need of explanation; many problems are overlooked and can not be solved adequately.

An overview of some established and recognized models of user-centered design process is given as follows:

The UCD process is formalized in the ISO-standard 9241-210 [ISO10]. The standard describes UCD as an iterative process consisting of five steps (see Figure 4.1) and as a multidisciplinary activity that combines the user with its expectations and requirements and ergonomic findings. Other objectives of this standard are to improve working conditions, increase productivity and effectiveness of work processes and to prevent adverse effects on health. It also introduces the following key principles:

- The design is based upon an explicit understanding of users, tasks and environments.
- Users are involved throughout design and development.

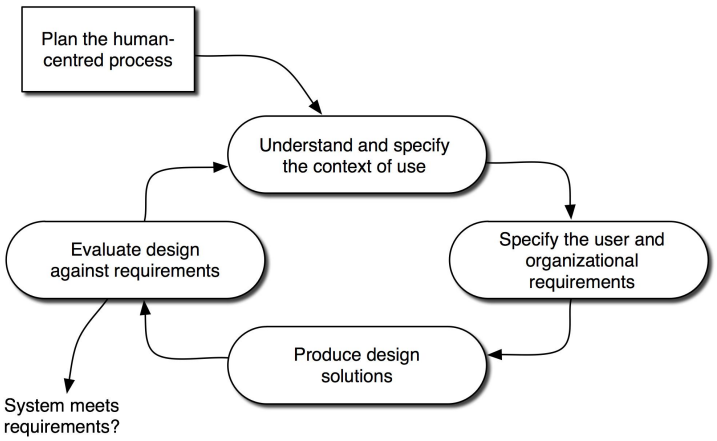


Figure 2.3: The User-Centered Design process, ISO-9241-210 ([ISO10])

- The design is driven and refined by user-centered evaluation.
- The process is iterative.
- The design addresses the whole user experience.
- The design team includes multidisciplinary skills and perspectives.

Nielsen [Nie93] describe the following stages of the Usability Engineering Lifecycle Model:

1. Know the user
   a) Individual user characteristics
   b) The user's current and desired tasks
   c) Functional analysis
   d) The evaluation of the user and the job
2. Competitive analysis
3. Setting usability goals - Financial impact analysis
4. Parallel design
5. Participatory design
6. Coordinated design of the total interface
7. Apply guidelines and heuristic analysis
8. Prototyping

9. Empirical testing
10. Iterative testing - Capture design rationale
11. Collect feedback from field use

It is not necessary to perform each step in order to achieve a positive result. In case of limited financial resources it is recommended to perform as many steps as possible before the design, so it is later not needed to change it because of new insights regarding the usability [Nie93]. As one of the first authors Nielson showed that Usability Engineering (UE) goes beyond testing and inspection of developed software, it is a systematic approach that begins before the actual development of the system and continues after completion of the system.

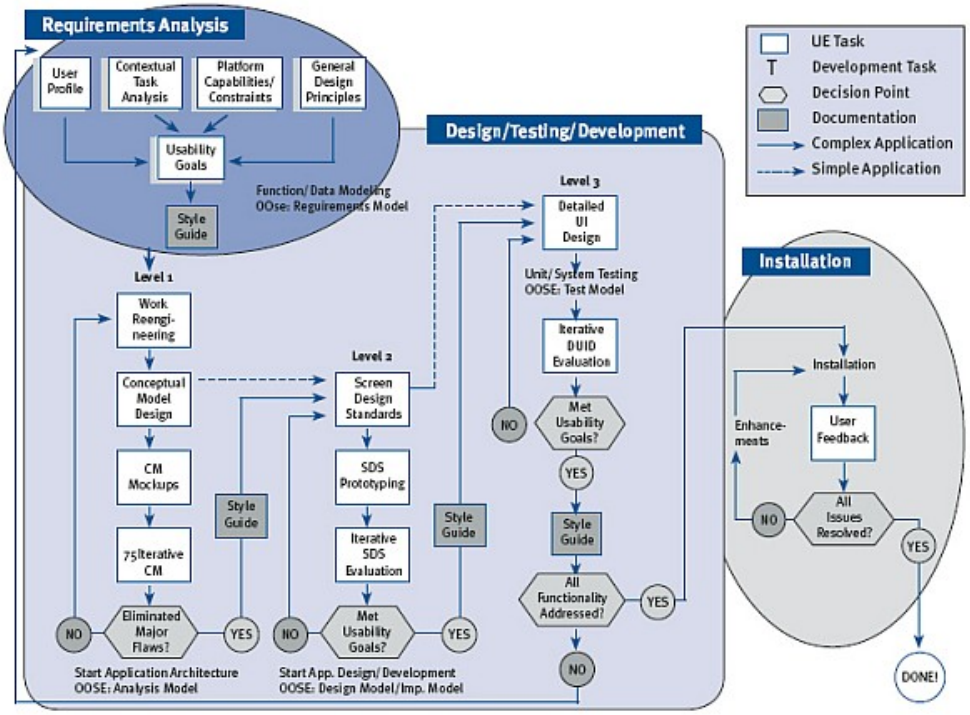


Figure 2.4: Usability Engineering Lifecycle by Mayhew ([May99])

Mayhew[May99] shows a structured and systematic approach to UE called the "Usability Engineering Lifecycle" and differs three main phases: requirements analysis, the design/evaluation/development and the installation (see Figure 2.4). In the first phase (Requirements Analysis) users and their tasks, as well as technical capabilities and limitations are analyzed. Along with general design principles, the findings are specified as targets of usability and provided in "Product Style Guide". The second phase (Design/Testing/Development) contains three levels. The first level involves the

organization of work and the development of a conceptual model. It is checked in an iterative evaluation, whether the objectives identified in the requirement analysis have been achieved. This is the starting point of the second level, in which prototypes are developed based on previous findings and evaluated iteratively. In the third level a detailed user interface is developed and reviewed iteratively until all requirements are met. Thereafter, the phase of installation follows, in which improvements based on user feedback are performed.

Beyer and Holtzblatt [Bey99] describe a design process called "Contextual Design", which focuses on the users in their respective context. It is suitable for large, complex as well as small projects. The procedure consists of six sequential steps, which are undertaken before the development process of the software begins. The steps are as follows:

- Contextual Inquiry
- Work Modeling
- Consolidation
- Work Redesign
- User Environment Design
- Mock-up and Tests with Customers

In the first step information about the user and their work is collected and then visualized in different models (Work Modeling). The step "Consolidation" merges all the information and models to a holistic image and gives an overview of the activities of the user. This holistic view is important to create the designs of the user interface in the next steps "Work Redesign" and "User Environment Design". "Work Redesign" aims to optimize the workflows of the users. In "User Environment Design" the general structure of the user interface is defined. This produces the so-called "User Environment Diagram". This forms the basis for the last step, in which mock-ups and prototypes are developed to validate the designs with the users.

Cooper et al. [CRC11] describe a method called "Goal-Directed-Design", which aims to design usable software systems. The focus of all design activities are the objectives of the user, which should be supported with the system to be developed. The method consists of five phases, which take place before the implementation of a system 2.5. The results of a phase are the basis for the next phase. Special emphasis is on the transformation of the analytical results of the "Research Phase" into a visual design in the "Refinement Phase". The procedure of the "Goal-Directed-Design" approach is comparable to that of the "Contextual Design". In both cases, a procedure is described, which starts with a thorough analysis and ends with the modeling of the design. But in "Contextual Design" the validation of users is described as an explicit step.

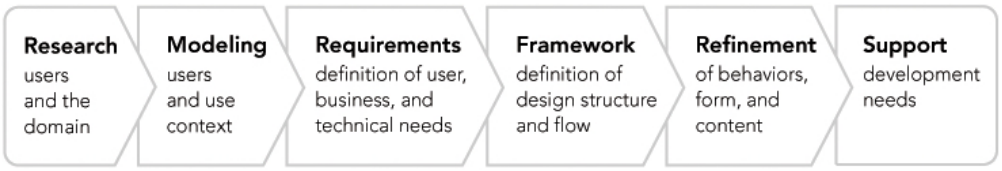


Figure 2.5: Goal-Directed-Design Process by Cooper et al. ([CRC11])

Preece et. al. [SRP07] suggest a simple lifecycle model for interaction design, which also covers iteration and focuses on user's needs. The model consists of four basic activities 2.6. The first activity targets to identify needs of the users and to establish requirements. These are used to develop alternative designs, including a conceptual design and a physical design of the product. Then interactive versions of the designs are built and evaluated. Based on the feedback from the evaluations, the team may need to return to identifying needs or refining requirements, or it may go straight into redesigning. It may be that more than one alternative design follows this iterative cycle in parallel with others, or it may be that one alternative at a time is considered [SRP07].

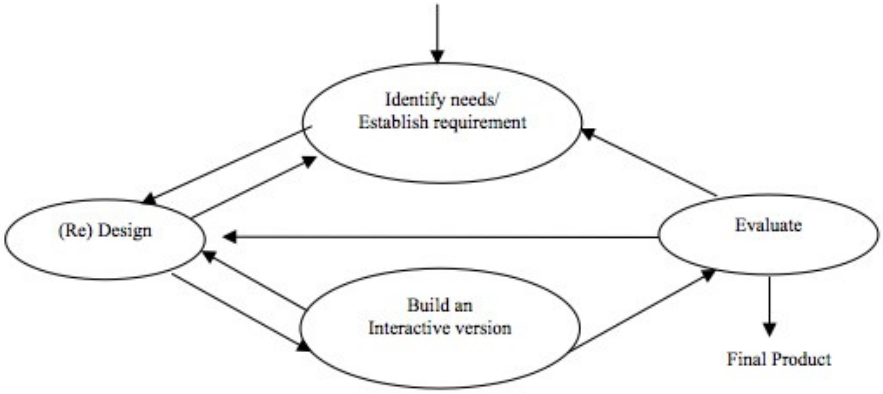


Figure 2.6: A Simple Interaction Design Model by Preece et al. ([SRP07])

The selected models shown here are different in detail and put their focus on different aspects, still they all describe the same procedure in principle: a user-centered design process. In almost all models, the rules of Gould et al. [GL85] can be found again: the call for an iterative approach, the early focus on user and task requirements and the empirical verification of the drafts by users.

2.4 Usability Engineering

The term UE was coined in the mid-1980s [Ben84, Gou88, Sha85, WBH88]. At that time developers considered it as their main task to make the technology faster, smaller and more powerful. However, the HCI experts put their focus on the usability components: effectiveness, efficiency and ease of use. In order to meet these requirements and optimize the usability of an application, it is necessary to involve the users and their needs in the development of user interfaces. This optimization can take place wherever users interact with the systems and thus a form of user interface is available. Optimizing the user interface is not enough to improve the usability of a system, a low error rate and a good performance are also important to increase user's satisfaction and to make the system more efficient [RF13]. The evaluation of usability as a quality feature and its improvement by the developer is usually done within a process, which is called UE. Thus, in addition to the traditional software development process also the process of UE is to be considered.

Nevertheless, this is far too rarely the case in practice. Richter et al. [RF13] explain this problem as follows: Although the consideration of user requirements in the current Software Engineering (SE) development model is regarded important, corresponding activity descriptions, methods and techniques are missing. The result is that many projects consider the inclusion of users as an option or as an additional effort and therefore proceed without corresponding activities. Another problem is that authors of process models widely address usability professionals. This creates the feeling that these approaches are traded as elaborate, independent disciplines, which leads to the impression that usability engineering is a time-consuming and cost-intensive extra.

However, according to Richter et al. UE can be seamlessly integrated into the common software development process. The use of usability methods in requirements management has proved to be succesfull. On the one hand the requirements of the users are systematically included in the analysis, on the other hand it can be ensured that these requirements have actually been realized. The aim of an integration is to combine the objectives and procedures of SE with those of UE so that a systematic and predictable implementation during the development is formed, which considers factors like cost, time and quality. There are several methods and systematic approaches for the support of integration and development. All of them have much in common since they describe an idealized approach that ensures the development of usable software, but they differ in their specifics, in the applied methods and the general description of the procedure (e.g. phases, dependencies, goals, responsibilities, etc.)[Wol]. Considering the focus of these approaches, four general categories of integration can be differentiated [Fis12]:

Concrete Implementations

Approaches of these category refer to the concrete implementation and define activities, results and their linkage with existing SE activities. They aim a gentle implementation of UE aspects at the level of a common base. Overall, these approaches focus on bringing as few organizational and structural changes. As an example, Schaffer [Sch04] introduced a method for integration of UE activities that are based on the evaluation and testing of an existing solution. The results are successively integrated into the process of development.

Specifications

These integration approaches are very similar to the approaches of concrete implementations. They rely on communication and ensuring the mutual flow of information and define for this purpose common notations and artifacts of the development process. Examples include the annotation of use cases with tasks descriptions for the extension of SE artifacts [CL99, Ros99], the complement of object-oriented SE notations and models [NeC00, DP01, Van01] or pattern-based approaches focusing on recurring patterns and corresponding user-centered and optimized software solutions [DLH02, JLMS03, Tid10].

Definition of Processes and Models

A third group of integration approaches address the definition of processes and process models distinguishing between independent UE models and SE models with integrated UE activities. Examples for independent UE models are the "Star Lifecycle" by Hix and Hartson [HH93], the "Task-Centered User Interface Design" by Lewis and Riemann [LR94] , the "Contextual Design" by Beyer and Holtzblatt [Bey99], the "Usability Engineering Lifecycle" by Mayhew [May99], the "Usage Centered Design" by Constantine [CL99] and the "Scenario-based Approach" by Carroll [Car00]. Some examples of approaches, which build on existing SE models are given as follows: Burmester et al. [BMS05] identify the conceptual differences between UE and the procedure model of Rational Unified Process (RUP) and provide examples of the integration. This is achieved by insertion of UE aspects into various phases and activities of the procedural model. Meinhardt and Beck [MB05] show how through the "Usability and Ergonomics Process Module" in the V-Model XT [HH08] the usability can be fully integrated into projects and what should be considered for a successful implementation. Düchting et al. [DZN07] analyze the agile process models "Extreme Programming" and "Scrum" in relation to the implementation of UE activities based on the DIN ISO 13407 and derive appropriate recommendations for the user-centered implementation in practice. Paelke and Nebe [PN08] extend these recommendations, derive an adapted procedure model for Scrum and show the application of the model with an example of mixed reality.

Abstract and Generic Approaches

The approaches of this category are detached from specific models or procedures, focus on abstract or generic approaches and describe organizational conditions, principles, paradigms and meta-models. Thus Pawar [Paw04] analyzed basic similarities in communication between activities of SE and UE, which resulted in a framework defining the exchange of information between activities of SE and UE and a corresponding coordination strategy. Granollers et al. [Gra] introduce a general process modell called "Usability Engineering Process Model" for a user-centered development process, which combines traditional practices of SE with prototyping and evaluation. Ferre [Fer03] presented a set of usability techniques which can be incorporated incrementally without large organizational effort in a development process. Sousa et al. [SFM05] describe another generic process model describing a software development process aiming at usability, productivity and integration. Some other examples for process models are the "Common Software Development Framework For Coordinating Usability Engineering and Software Engineering Activities" from Pawar [Paw04], "Method-Independent Process Model of User-Centred Design" from [Jok02] and the "Evidence-Based Computer-Aided Usability Engineering Environment (CAUSE)" from Metzker et al. [MR02].

All approaches have in common to contribute to the systematic approach of developing usable products. Important is the consideration of the perspectives of both disciplines. Depending on the focus, existing approaches address three different levels of abstraction [NZP08, NP11]:

- the abstract, superordinate level of standards that serve as a framework for ensuring and maintaining consistency and quality across within and across borders of an organization,
- the level of models, which serves as a template for the implementation, and
- the operational level in which the models are tailored to fit the specifics of an organization.

As explained in the chapter 2.3 the discipline UCD, is concerned with how users can be systematically involved in the development of systems and products. As a central process model of UE, UCD has prevailed.

2.5 Usability Methods

It is important to consider the user and how to make them an active part of each step in the design process. There are several methods to involve the users in the design process. In the following the methods are explained with reference to the phases of the UCD process 2.3.

2.5.1 User & Requirements Analysis

In this phase information about the user interface, the users, tasks and the context of use are collected in order to determine the requirements to be formalized. An analysis is essential to be able to respond to the different needs. However, it is not excluded that some requirements change throughout the process or new requirements arise. The following methodologies are usually used at this stage:

Questionnaires and Interviews

Questionnaires are the most widely used method to collect data [Tah16]. The aim is to obtain information, assessments and ratings of certain individuals on the system. The target audience can be the current or potential users of the systems, experts or developers. A questioning may be conducted orally or in writing. The difference between oral and written questioning is the situation. A written questioning is more anonymous than an oral, which may have a positive effect on the honesty of statements. In a written questioning, however, it may not be clear whether the default order of the questions has been followed and whether the person answered the questions, who was addressed. A written questioning is performed in form of questionnaires or checklists, whereas the oral is performed in form of interviews.

In contrast to questionnaires, interviews are a method of qualitative research. Basically, a distinction can be made between closed or semi-open interviews and open interviews. In a closed interview the questions and answers are given. The interviewer takes an active, authoritarian role. However in open interviews there are no fixed questions and answers, but a guide. The interviewer steps into the background and influences the course of the conversation by using guiding questions [BD06]. Quantitative and qualitative surveys are not fundamentally different research methods but the two ends of a continuum. The results of a qualitative study are primarily aimed on understanding, not just describing [Lan].

Contextual Inquiry

Contextual Inquiry is a method developed by Beyer and Holtzblatt[Bey99], which is based on an "Master-Apprentice-Modell" of learning: observing and asking the user questions at work as if he were the master and the interviewer the apprentice. The aim is to gain a solid understanding of the needs, desires and attitudes of the users. Concrete tasks, procedures and patterns of behavior and the environment of the application are analyzed, evaluated and documented. The interview may take place in written or oral form. Unlike classic interviews, the interviewer is guided by occurring events and observations.

It is of critical importance that subjects be interviewed in the places where they actually use the products. Not only does this give the interviewers the opportunity to witness the

product being used, but it also gives the interview team access to the environment in which the interaction occurs. This can give tremendous insight into product constraints and user needs and goals [CRC11].

Card-Sorting

Card sorting is a method that can be used at any time if feedback on the content, terminology and organization of the product is desired. The basic principle of this method is to provide current or potential users cards with items. The users group the items individually into clusters. The clusters may provide information about optimal number, names and scope of categories and depth of hierarchies.

2.5.2 Design & Implementation

In the design phase concepts regarding the system to be designed are created. The actual implementation of the system is performed in the development phase. In the following methods are described that are frequently used in these phases.

Prototyping

Prototypes provide a good basis for UE and can be evaluated with usability tests. Aspects of the user interface are designed, to further refine the behaviour of the system. The prototypes visualize the current application and serve as an easily understandable and common language between users, customers and developers.

As a first form of prototype sketches of the user interface can be used to identify criticisms at an early stage (also referred as Paper-Prototyping) [Nie93]. Some other representatives of prototyping are "Rapid Prototyping" and "Interactive Prototyping". "Rapid Prototyping", which is a technique for rapid development of prototypes usually makes use of software tools for graphical design development. "Interactive Prototyping" allows immediate manipulation of the prototype in cooperation of the user [Pre13].

Cognitive Modelling

Cognitive Modelling is a useful method for estimating the efficiency of a user interface. The aim is to make as thorough and precise predictions about the expected time and cognitive effort of a skilled user while the user goes through specific interaction tasks. Such predictions can facilitate the work of interaction designers considerably when it comes to compare some design variations with each other. Some well-known ways for Cognitive Modeling are GOMS [JK96] and CogTool [SH10].

Keystroke-Level Model (also called Action Analysis)

In this method the actions of a user are simulated and the keystrokes for a certain action are recorded. It involves breaking the task into individual actions such as move-mouse-to-menu or type-on the-keyboard and calculating the times needed to perform the action [Hol05].

2.5.3 Evaluation

Inspection

This category refers to a number of methods that are based on the judgments of experts. Experts can be domain experts and software developers, but also users with great experience in dealing with the system to be examined. Some examples of methods involving experts are:

- Cognitive Walkthrough: Regards primarily issues of learnability of a system [Gil09]. In this approach, the expert first checks whether the user can form reasonable goals and then checks whether the user can perform appropriate actions in order to achieve its goals. The expert breaks the goal down to steps, which are required to successfully reach it. At each step, the expert asks among other things, whether the correct action can be carried out and leads to the desired effect. A negative answer to a question may indicate a possible usability problem. To identify likely problems, the experts have to make judgments based on the overall impact of the potential problems for a task.

- Heuristic Evaluation: A small number of experts assess the usability of a system using heuristics. Nielsen [Nie92] recommends 3-5 experts to detect serious usability problems.

- Focus-Group: Moderated discussion in small groups are conducted [Mor97] to explore perceptions, opinions and assessment of users.

- Guideline-Inspection: The implementation of a user interface can be supported by design guidelines. Usability guidelines can be seen as an appropriate basis for designing User Interfaces. During a Guideline-Inspection the compliance of an interface with a detailed list of usability criteria is reviewed. According to Mike and Nielsen [MN95] this method can be regarded as a kind of cross between a heuristic evaluation and a standard inspection, because it contains elements of both methods.

Usability Testing

In this method the users test the application by performing predetermined tasks. It is according to Nielsen the most fundamental usability method and is irreplaceable, since it provides direct information about how people use computers and what their exact problems are with the concrete interface being tested [Nie93].

- Thinking-Aloud: The test participants are encouraged to express their actions and thoughts during learning aloud. Moreover, they also verbalize expressions of satisfaction, excitement and motivation. The aim of the method is to ascertain the chain of thoughts of the users. The major advantage of this method is that the test persons reveal not only usability problems, but most also explain why they have these. However, this method also has drawbacks. It may happen that the users forget to think aloud, because they have to focus on two tasks, firstly on using the new systems and secondly on thinking aloud.
- Question-Asking Protocol: As in the Thinking-Aloud method the users verbalizes his thoughts while using the systems. The difference is, that the expert triggers the users verbalizing their thoughts by asking direct questions.
- Field Observation: In this method by Nielsen [Nie93] the expert goes into the real environment of a user, to watch him while working with the system. The observers should interact little with the user, so that as a true reflection of human-machine interaction can be obtained. The expert notes his observations or, if possible, records the situation on video. After the observation a survey can be carried out.
- Constructive Interaction: It is a variation of the thinking-aloud method, in which two users work together to solve a task. Therefore users may make more comments than in the traditional Think-Aloud method. Nevertheless, the method has the drawback, that the users may have different strategies for learning and solving tasks. Therefore the test session may switch back and forth between different ways of using the interface and one may also occasionally find that the two users simply cannot work together [Nie93].
- Retrospective Testing: In comparison to the thinking-aloud method, the participants of the test work without expressing their thoughts during the test. Instead the test is recorded on video. After the test, the participants make their comments on the basis of video recording. Thus one wants to benefit from decrease of the reactivity of the participants by letting them solve the tasks in their own manner and pace [vdHdJS03].
- Coaching: The test participant is allowed to ask any system-relevant questions to the coach (expert). The expert has to answer the questions [Kra13]. The purpose of this technique is to discover the information needs of users in order to provide better training and documentation, as well as possibly redesign the interface to avoid the need for the questions [Nie93].

- Shadowing: During a usability test, the user is observed by an expert user (shadow), who tries to explain the behaviour of the user to the tester.
- Teaching: A user attempts to become familiar with the system in a given time and explains later another user the functionality of the system.
- Wizard-of-Oz: Complex functions of a system are simulated by a developer without the test person noticing [Kra13].
- Eye-Tracking: The method captures the eye movements of the user during the use of the system using an 'Eye Tracking Camera'. The aim is to find out on which areas of the screen the user focuses.

2.6 Style Guides

Style guides (also referred as Web Style Guide, UI Style Guide or Front-End Style Guide in context of a website or web application) are a structured collection of principles, guidelines and recommendations for the development of user interfaces and aim a unification of the user interface (see Figure 2.7). They do not only contain abstractly formulated requirements for the design of navigation, buttons, menus, etc. but also code snippets or patterns for the operational use by the developers of the application.

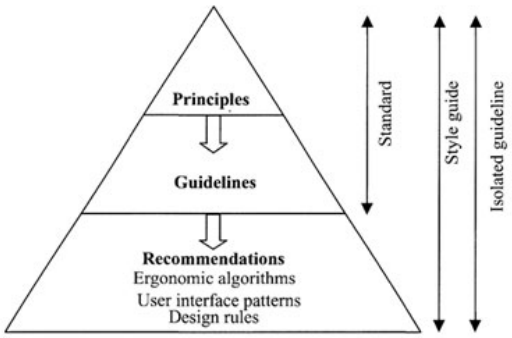


Figure 2.7: Classification of guidelines [MVP04]

In many projects the developers of the application design and implement the user interface, but often have difficulties in taking design decisions [JK05]. They usually do not have special training in the area of interaction design and would like to get better support [OBW01]. Usability experts are involved in the design of the user interface in some cases, but hardly ever in the actual implementation of it [Met01]. Without systematic documentation of the design, deviations occur, which harm the recognition value and the usability of the product. A style guide is a simple and binding guideline for all those who are involved in the development. Many companies introduce style guides to establish a

usability engineering process [HR12]. They are often regarded as a repository for user interface related policies. Style guides are never complete and finished, that means that they need maintenance and updating in each phase of the software development process. Thus they are not only an input for the development process, but also a part of it. A collaboration and development platform has to be provided, with which the participants can actively influence this development [RH13].

Usability Guidelines

Guidelines exist in different forms and can be found in numerous sources. They can be classified on the basis of their type and origin. According to Mariage [MVP04] there are the following forms:

- Principles: affect the conceptual design decisions during the development process and reflect the knowledge of human perception, learning ability and behaviour of people. They are formulated generally, so that they can be used for a wide range of use cases.
- Guidelines: are based on principles, but are concerned with a particular area of design. However, they always need an interpretation of the guideline in order to satisfy the needs of an organization or a special design matter.
- Recommendations: define a conceptual decision for a specific range of application. They are clear and leave no room for interpretation. e.g. "Provide search, help, and home page links at the bottom of each page." Such recommendations are derived from existing principles and guidelines taking into account the requirements and context of use. The recommendations include design rules, ergonomic algorithms and user interface patterns.
  - Design rules: consist of a set of functional requirements and specify a constant design for specific interfaces. There is no need for further interpretation by the developer or designer.
  - Ergonomic algorithms: use usually simple design rules in a systematic procedure which is faster than single applicable rules. They aim to systemize design aspects of a user interface.
  - Interface patterns: are a concept to combine multiple isolated guidelines into an understandable and applicable system of rules. They offer a comprehensive solution for a whole range of similar design problems through various websites and applications.
- Standards: Another type of guidelines are standards. They aim to ensure a standardization of user interfaces. Publisher are national and international organizations such as governments, military and civilian or industrial facilities.

Figure 2.7 illustrates the relation between the different types of guidelines.

User Interface Patterns

User Interface Patterns or Design Patterns group several isolated guidelines into one understandable and applicable design rule. The difference between patterns and guidelines lies primarily in the perspective and the representation of information. User Interface Patterns describe frequently occurring or similar design problems in patterns and offer tried and tested solutions. In contrast to guidelines, patterns have the advantage of being needed in a context of use. The users of a guideline often do not know the problem at which a certain guideline is to be applied [MVP04]. In the literature often the use of patterns rather than guidelines is recommended [MVP04, VM, DR13, GP05]. In practice the delimitation to elementary collections, as often used in style guides, is often not clear. In this work, no distinction is made between guidelines and patterns, as the concepts are considered to be interchangeable in the context of the work. Consequently, the term guideline is used representatively for both.
Figure 2.8 shows an example of a guideline and an user interface pattern.

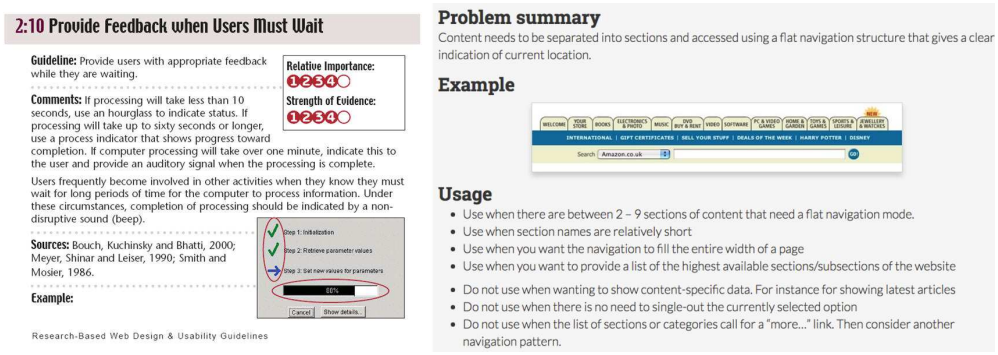


Figure 2.8: Left: a guideline (Usability.gov, 2017) - Right: a design pattern (ui-patterns.com, 2017)

2.6.1 Types of style guides

Richter et al. [RF13] classified style guides in three different types, with different scopes.

Platform Style Guides

Platform style guides are style guides, which describe specific design requirements for a certain application platform. They have mostly been developed by hardware and software manufacturers - such as Apple, Microsoft or IBM - to enforce industry standards and contain operationalized information regarding the design of windows and dialogs. IOS Human Interface Guidelines [Exp11] and Java Look and Feel Guidelines [Sun99] are some examples of style guides of this category.

Company Style Guides

Company style guides give specifications for the implementation of a company-specific look & feel. In addition to company style guides intended for the use by external customers, style guides are often used for company-internal development processes. The goal is to ensure ergonomic design and consistency of the user interfaces of internal projects.

Project Style Guides

None of the types of style guides mentioned above form a perfect solution for creating a corporate application style guide. Platform style guides are too detailed, so that project teams working in a familiar environment spend too much time on platform basics. Company style guides are too general and usually do not provide answers to basic design questions. Project style guides meet the necessity of a style guide, which is a hybrid of platform style guides and company style guides. They contain guidelines to ensure the consistency of the user interface when developing an application or a product for the end user. In addition to providing guidelines, a project style guide also serves an educational purpose. Beside illustrating the conceptual approach to the user interface, it also may contain a process to ensure that usability is built into the design from the beginning [Que01].

Frost [Bra14a] goes one step further and divides company and project style guides according to their usage. This results in style guides for branding, design language, voice and tone, writing, patterns and coding. Based on these types, Debenham [Deb] differentiates the following main types of style guides:

Content and editorial style guides

Content and editorial style guides provide guidelines for the content of a website or publication to ensure consistency. MailChimp's seminal voice and tone [Mai] website is an example of such a style guide.

Brand identity guidelines

The most commonly used type is the brand identity guideline, which describes the basics of a brand. Among other things, the brand identity guideline often includes the company's logo, the limitations associated with its use and the brand colors.

Code conventions

In order to keep the code maintainable and clean, conventions, patterns, and examples are given.

Human interface guidelines

Human interface guidelines provide interaction guidelines with the goal of standardizing interactions with a given product and not confusing users with inconsistent responses to interactions.

Front-end style guides

While the purpose of a brand identity guideline is to maintain the brand consistency, the front-end style guide (also referred as UI style guide) is used to maintain the consistency of the website or the application. It documents the visual appearance and the functionality of a website and acts as a tool for communicating and maintaining the user interface. Unlike a pdf or PowerPoint based style guide, a front-end style guide consists of HTML, CSS and JavaScript files and can be viewed in a browser. It is interactive and can also be living, which means, that it references the same CSS file as the website it is designed for. As a result, a change on the website automatically affects the style guide [Deb]. Ideally, the HTML and JavaScript files are the same, but this is not always possible due to the structure of the website [Fri13].

2.6.2 Benefits of style guides

The use of a style guide has a wide variety of benefits. The style guide can be used as a communication tool within the development team and with stakeholders. The vocabulary used in the style guide forms a kind of terminology for the components of the website. This facilitates the communication with the client and other stakeholders. Within a team, a style guide can provide information about existing code to avoid rewriting code [Fro].

In the case of projects, which have a long development period or many team members, design decisions are quickly forgotten. The documentation within a style guide provides an important communication basis for discussing and coordinating tasks, preventing unintended information loss and reducing errors. In addition, tasks or processes that have already been executed do not need to be repeated completely, i.e. if new team members are assigned to a project. Another advantage is that design decisions within a style guide can be reused in similar projects and thus minimize the effort put in future projects [RF13].

If the style guide serves as a driver for the design and implementation, it is more likely that the resulting website matches in terms of appearance and code. The designer can check the basic elements of the site, such as colors and fonts, as well as the available components and make sure that the new design fits well with the overall design of the website. The number of different components can be reduced by the consistent use of available components from the style guide. The integrity of the interface increases the credibility of the business [Fro].

A well-functioning, style guide driven development anticipates the development of the user interface. Because the design of the user interface is made up of components, the development of the individual components can begin, even if the overall design or the actual content, such as text and images are not complete. This can be used to obtain feedback from different stakeholders such as the client or the user of the website, and to respond to potential change requests earlier in the project [Lew14].

Debenham mentions, that a style guide can also be used to speed up component testing with different browsers by testing parts of the user interface in the style guide [Deb].

Gale addresses an extensive list of benefits from different perspectives (see table 2.2).

End user	Development	Business
Reduced errors - less frustration	Maintain control over look and feel	Produce usable systems enhancing customer service/satisfaction
Increased confidence in the system	Control 3rd parties during the tendering and implementation phases	Increase market awareness
Reduced training requirements	Reduce arbitrary design decisions and reinvention	Increase product awareness
Increased morale	Capitalise on learning/iteration	Reduce training costs
Improved use of system functionality	Enable production of reusable software	Facilitate helpline support
Improved productivity	Reduce development time	Improve staff retention
Reduced resistance to the use of new technology		Increase user acceptance of new systems
Reduced training requirements		

Table 2.2: Benefits of style guides seen from different perspectives [GS96]

Quesenbery [Que01] defines usability goals for project style guides and refers to the definition of usability according to ISO standard 9241-11 [ISOa] (see table 3.1).

Usability Characteristic	Goal
Efficient	Improved Quality: The time required to design of the user interface will be reduced because basic guidelines are clearly documented, tools are shared, and best practice guidance is available for other decisions [Que01].
Effective	Improved Process: User interface will be able to work together better because shared design guidelines are available. Initial designs will be more effective, with less re-work to solve usability problems required [Que01].
Satisfying	Improved Usability: The user experience will be improved, both for the designers and users. Designers will have the satisfaction of creating excellent interfaces, while users will benefit from increased usability [Que01].

Table 2.3: Usability Goals For Project Style Guides ([Que01])

2.6.3 Criteria for success

Quesenbery [Que01] adresses the problem that style guides are frequently created, but rarely successfully used. Vogt [Vog01] identified some difficulties in using style guides. These include, among other things, the large scope and the resulting confusion, a large depth of detail so that the underlying ergonomic considerations are no longer identifiable, contradictions within style guides, as well as the danger of viewing visual consistency as the sole determining usability feature. Kuniavsky et al. [KR05] emphasizes to consistently take the viewpoint of the developers. The development of a style guide has to be considered as a process, in which developers represent the addressed user group. Therefore, it is important to understand the developers and their tasks to provide support that is optimally suited for their needs. Gale [GS96] also explains the low acceptance of many style guides by not adequately observing this user-centered view. The fast and successful search of relevant information is one of the most important criteria from the development perspective. Therefore, Wilson [Wil04] identifies an insufficient index as one of the most common acceptance problems of style guides. Frost [Fro] emphasizes in this context the importance of creating a style guide, that looks appealing and is easy to navigate.

There are numerous other sources in the literature with reasons that can ultimately lead to a low acceptance of style guides. However, various authors also offer suggestions that can be used to solve these problems. Table 2.4 summarizes and categorizes suggestions of different authors and has been used as a checklist for the development of the online style guide.

Name	Description
Start early	Style guides should be created early in the development process, in order to involve all relevant stakeholders of a design. [Que01, GS96].
Raising the awareness of developers and end users	To understand the meaning and purpose of the style guide by the developers and end users, they should be sensitized in collaboration with the management [GS96, Wil04].
Make the process transparent	Comments and discussions should be integrated into the style guide so that stakeholders can follow the progress of the work [Que01, GS96].
Make the context visible	It should be clear, in which context a style guide should be used, what its purpose is and not is [Que01, Vog01].
Make the emerging work widely available	A style guide should be made widely available during its development. This allows users to influence in the form of comments and questions. However, to ensure that unfinished style guides are not used prematurely, they must have a clear status [Que01, GS96].
Integration into the development process	A style guide should take into account the development process, in which it is intended to be used [GS96, RH13].
Notification of the users about changes	A method should be offered that allows users to be aware of any changes [Wil04].
Define useful guidelines	It should be clear what guidelines have to be followed and which are only a recommendation [Wil04]. Furthermore it should explain why it promotes usability and what its purpose is [Que01].
Ensure consistency of examples	Ensure that all the given examples are consistent with the corresponding guidelines [Wil04].
Use negative examples	Negative examples show what should not be done. They have to be prepared with care, since they must be conform to the entire style guide and not just to to the particular guideline they illustrate. It has to be clear whether the given example is a negative example or not [Vog01].
Connect guidelines to analysis	If there are results from user analysis or usability tests, that confirm a guide line, they should be integrated into the style guide [Que01].
Limit scope	A style guide should restrict itself to truly important areas and should not treat every single topic. Therefore, it is a good idea to develope style guides for a particular purpose. Rather than describing known components of a user interface in detail, style guides should address their context of use and combination with other components [Wil04, Vog01, GS96].
Training and support	Style guides should also introduce educational input [GS96, Que01].
Put it online and make it interactive	It should be planned from the beginning to put the style guide online and to make it interactive [Que01, RH13, GS96].
Right indexing and cross-referencing	A style guide should provide a useful index. Use hyperlinks to link content and examples. Topics should be organized in such a way, that the most important information is at the beginning. Each topic should only be dealt within a single hypertext page and refer to other topics using hyperlinks. You should take time to create an index and if possible use the help of a professional indexer [Que01, RH13].
Create role-specific views	If possible different contents should be prepared depending on the role of the current viewer [RH13]
Allow further development of the style guide	A style guide is not a static document. It is therefore important to create an environment, in which it is possible to improve a style guide over time [GS96].
Ensure responsibilities	It must be clear who is responsible for maintaining and updating a style guide. This can be a group or a single person [GS96].

Table 2.4: Checklist summarizing recommendations for creating style guides

2.6.4 Content of style guides

There are also various recommendations on the content of style guides in the literature. Gale [GS96] and Quesenbery [Que01] give examples of possible outlines for style guides. Richter et al. [RF13] identifies the following information as necessary:

Content of project-specific Styleguides	
Technological framework and target group	Describes to which systems the style guide refers to and who the target group is.
Software ergonomics	General rules to be considered in the specific case (e.g., number of menu items) as well as rules relating to the specific target group and application.
Rules of application	Describes which elements to use in which situations.
Behavior of the GUI elements	Description of the system response, e.g. selection of an entry, deactivation of controls, etc.
Navigation	Description of the navigation elements, e.g. use of menus, links, buttons, etc.
Visual design	color scheme, contrasts, fonts, layout, distances, icons etc. Corporate design aspects are also referenced.

Table 2.5: Content of project-specific Styleguides [RF13]

Laubheimer [LAU16] gives a list of 25 common UI components included in web style guides and defines the following checklist of 8 main requirements:

- Table of contents that splits components into easily-findable categories
- Responsive layout or grid systems used to place common UI elements
- Color palette of the product (in appropriate format, e.g. HEX for the Web, UIColor for an iOS app)
- Typeface styles (e.g. H1 Title, Body text, Photo caption text), which should include typeface name and foundry, sizes, weights, leading/line height, tracking/kerning, and the appropriate contexts of use for that text style

Then, for each of the specific UI elements following information is to be included:

- Description of the appropriate context of use: When does it make sense to use one particular component versus a similar one?
- Code snippets
- Specs for implementation, including positioning and spacing information
- Dos and don'ts for that element.

Frost [Fro16] and Friedman [Fri13] provide some recommendations on the content of a single component in a component library. Thus, the following information can be provided for a component:

- The component name
- A description of the component
- Operating instructions and sources for the component
- A living example of the component
- Copiable code sample of the component
- Separation of component-related codes such as HTML, CSS and JavaScript
- The usage context of the component
- Information about the authors of the component
- Component metadata

Naming and viewing a single component in a style guide is often important for communication. A uniform naming of the component makes it possible to reduce misunderstandings and simplifies communication between the stakeholders. In addition to the name, a brief description of the component and its intended use can be displayed. In some situations, it may be necessary to add a separate, extended manual to the component that describes the options and limitations associated with the use of the component. The manual may also include instructions for operating a component and links to external sources. The design and the behavior of a component can be illustrated by a live example, for which also the HTML markup can be displayed, so that it can be copied if needed [Fri13, Fro16]. According to Frost [Fro16], the code associated with the component should, if possible, be separated to CSS and JavaScript codes.

Related components may be separately displayed in a style guide, so that the actual place of use on the website or web application can be difficult to understand. At this point, information about the context of the component is helpful. The context can be displayed through manually captured screenshots or dynamic links that can be used to navigate between the components in the style guide. Furthermore, the names of the persons, who have planned and developed the components can be provided. This information facilitates the maintenance and the further development of the components [Fri13]. A component may also be extended with useful metadata, e.g. status information showing, whether the component is incomplete, ready, or obsolete. In addition, component related version information like the version number and the latest change can be shown [Fro16].

In addition to the component library, the style guide can also contain guidelines for

writing code and content. These include, for example, the size of the indents used in the code, the naming of the files and the commenting of the code. A style guide can also provide other content pages, such as a news section, which displays the latest style guide changes [Deb].

2.6.5 Style guides in the software development process

A style guide can be maintained at various stages of user interface development. Coyier [Coy15] identified four models for maintaining style guides.

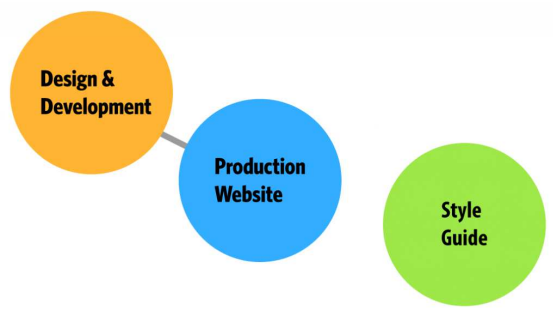


Figure 2.9: The sidelines model by Coyier ([Coy15])

The style guide can be completely separated from the website and its development process (see 2.9). In this case, the style guide has its own code base and is developed as a stand-alone unit, which challenges the timeliness of it. Since the style guide is separate from the website, it can not be updated automatically. Thus, updating the HTML tags in the style guide's live examples may be forgotten.

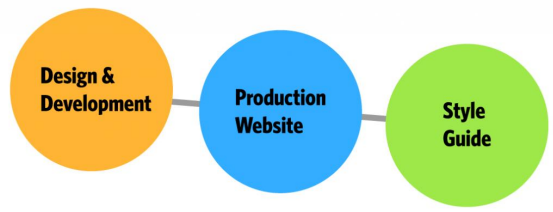


Figure 2.10: The exhaust model by Coyier ([Coy15])

On the other hand, a style guide may share part of its code base with a website so that it is updated when the website is edited (see 2.10). However, this does not promote component development, because the website is being developed overall. In addition,

there is a risk that the style guide loses on focus and attention, because of its placement at the end of the process.

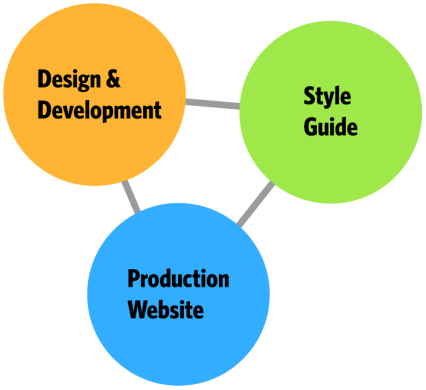


Figure 2.11: The colony model by Coyier ([Coy15])

Another way is to develop the style guide and the website simultaneously (see 2.11). In this case, the website gets updated when editing the style guide and vice versa. The style guide and the website use the same code base or the update is automated.

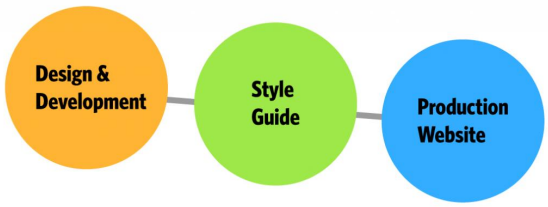


Figure 2.12: The dictator model by Coyier ([Coy15])

The style guide can also be seen as a driver for the development (see 2.12). No change is made to the user interface of the website before putting it into the style guide. The development of the user interface is initially in the style guide and not in the actual application, so that the style guide is always up to date. The frontend development can be separated from the backend development. Thus, the frontend development work can start earlier in the project. Once the background logic is complete, the user interface implementation from the style guide can be copied to the appropriate environment [Lew14].

De La Cuadra divides the style guide driven development process into four stages, which are shown in 2.13. According to De La Cuadra, the user interface development process starts with identifying the problem that needs to be resolved and the requirements captured. The designer or the developer attempts to design the new functionality based on the visual direction and guidance provided in the style guide. In some cases, functionality can be built using an existing component or by combining components, but sometimes new components must be implemented. The purpose, however, is to use the style guide as the starting point for the design and not start the design work from the beginning.

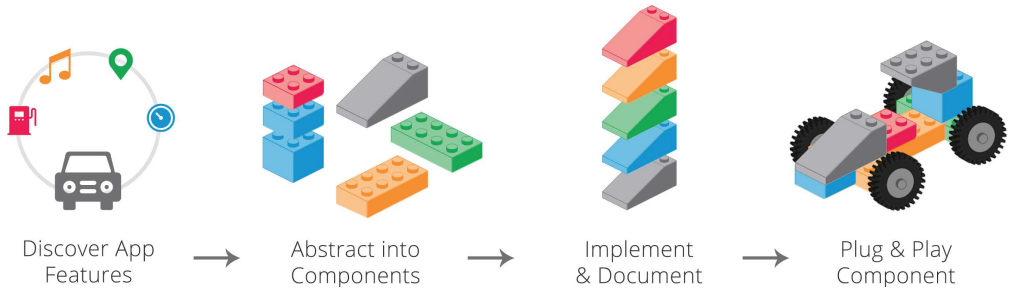


Figure 2.13: The stages of a style guide driven development process ([DLC15])

The created design is split into smaller sections to identify both, the components already available and those for which new developments are required. It is important that the designer and the developer go through the design together, because the static design does not always contain all the elements that are related to the layout and its function, e.g. animations. An interactive style guide can ensure that both have the same view of the issues being discussed [DLC15]. The components identified in the previous step are created in the style guide and separated from their context and other components. The separation of the components encourages developers to create components that can be used in different sections and for different purposes of the site [DLC15]. Once a component has been created in a style guide, it can be integrated into the actual website. A correction of the component must first be done in the style guide and then transferred to the website.

2.7 Usability of mobile applications

Mobile Usability is an emerging area of research, since it is considered to solve a variety of challenges, due to the unique characteristics of mobile devices. Considering the different constructions and the additional components of a mobile device, it quickly becomes clear that the usability requirements must be adapted to them. Adipat et al. [AZ05] identified several points of mobile applications different than desktop applications:

Name	Description
Environment	Users can be disturbed by their immediate environment, e.g. noise, other people or surrounding objects draw the user's attention.
Connectivity	Mobile devices have a slow and not always available Internet connection in some regions.
Small Screen size	The limited screen size can have a significant impact on the usability of mobile applications.
Different Display Resolution	There are many different mobile devices on the market that differ dramatically in the display size. Therefore, the usability test results can considerably vary.
Limited Processing Capability and Power	The power and storage capacity of mobile devices is far behind that of desktop computers. Therefore, some applications may not be optimally realized on mobile devices.
Data Entry Method	Small displays, virtual input methods, light irradiation or the user status (sitting, walking, running, etc.) can complicate data entry.

Table 2.6: Differences of mobile applications in contrast to desktop applications

Nielsen and Budiu [NB12] conducted usability studies to show the importance of optimizations of web applications for the mobile context. Thus, the probands used the mobile and non-mobile-optimized pages on their smartphone. As a result they identified higher success rates and more satisfied users with the user experience of the mobile sites.

Inostroza et al. [IRRR13] adapted the 10 Heuristics of Nielsen and developed 12 heuristics for mobile applications (see table 2.7)

Name	Description
Visibility of system status	The user should always be informed about the state of the system through feedback and in a reasonable time.
Match between system and the real world	The presentation of information should be based on the user and his environment. The vocabulary used should be adapted to the knowledge of the user.
User control and freedom	The user should be able to easily undo or redo his actions. These options should be preferably through a physical button or similar.
Consistency and standards	The device should follow the established conventions, on condition that the user should be able to do things in a familiar, standard and consistent way.
Error prevention	A good surface design should protect against potential mistakes. For this reason, for example, only the options available at one time should be displayed.
Minimize the user's memory load	Objects, actions and options should be visible. Users should not have to remember everything from one dialogue to another.
Customization and shortcuts	The user should have some configuration options to incorporate his personal preferences.
Efficiency of use and performance	The device should be able to load and display the required information in a reasonable time and minimize the required steps to perform a task.
Aesthetic and minimalist design	The device should avoid displaying unwanted information in a defined context of use.
Help users recognize, diagnose, and recover from errors	The device should display error messages in a language familiar to the user and offer a constructive solution.
Help and documentation	A step-by-step guide for the current task of the user should be available.
Physical interaction and ergonomics	The terminal should constantly provide buttons for the most important options. The positioning of the surface elements should be adapted to the ergonomics of a human hand.

Table 2.7: 12 heuristics for mobile applications

Name	Description
Suitability for the task	The dialogue should help the users to perform their jobs effectively and efficiently.
Self Descriptiveness	Each individual dialogue step should be immediately understandable through feedback from the system or is explained to the user on request.
Controllability	The user must be able to start the dialogue process and influence its direction and speed.
Conformity with user expectations	The dialog should be consistent and correspond to the characteristics of the user.
Error Tolerance	The intended work result should be achieved despite erroneous inputs either with no or minimal corrective action by the user.
Suitability for individualisation	The user should be able to adapt the dialog system to the requirements of the work task as well as to his individual abilities and preferences.
Suitability for learning	The user should be supported and instructed in learning the system.

Table 2.8: Sub-criteria for usability by ISO 9241-210 [ISO10]

2.8 Usability Criteria

Usability has to be reviewed with respect to its context and purpose; general, cross-context solutions do not exist. However there exist some criteria, which can serve as valuable indications for designing usable systems. According to ISO 9241-210 [ISO10], in addition to effectiveness, efficiency and satisfaction, usability is described by the following sub-criteria: Shneiderman defined golden rules of interface design based on his experience before the publication of the ISO standards and refined them over decades [SPCJ13] (see table 2.9). Another example of criteria defined based on experience is from Nielsen [Nie94]. According to Nielsen they are the minimum standards of usability an application must meet. These heuristics are given as follows: The 8 golden rules or the 10 heuristics reflect many aspects of the ISO criteria, which are shortly listed as follows:

- "Strive for consistency", "Cater to universal usability" and "Match between system and the real world" are similar in content to "Conformity with user expectations"
- "Offer informative feedback" and "Visibility of system status" are similar to "Self Descriptiveness"
- "Permit easy reversal of actions" and "User control and freedom" are similar to "Controllability"
- "Prevent errors" and "Help users recognize, diagnose, and recover from errors" are similar to "Error Tolerance"
- "Match between system and the real world" and "Flexibility and efficiency of use" are similar to "Cater to universal usability"

Name	Description
Strive for consistency	Related elements must remain consistent in layout, color and functionality throughout the application. Thus, the user does not have to constantly learn new operating concepts and the operation of a website is easier. Examples are the window bar and the navigation bar.
Cater to universal usability	There are different types of users whose different needs should be recognized and considered. They differ, for example in ages, knowledge, experience or disabilities. The design of a website should meet all types. While the site ensures easy operation for newbies, it provides keyboard shortcuts and shortcut menus to experienced users for quick operation.
Offer informative feedback	There should be feedback for each action. Feedback from the system is very important to a user to see if a particular action has produced the desired result. Small and frequent actions should trigger less feedback, whereas cases that occur rarely should be described in more detail.
Design dialogs to yield closure	Every function and every process has a beginning and an end. The user has a constant overview. For example, e-commerce web sites move users from selecting products to the checkout, ending with a clear confirmation page that completes the transaction.
Prevent errors	The interface should already be designed so that it is hardly possible to make erroneous entries. If an error nevertheless occurs, the error message should be as constructive as possible and suggestions for correcting the error should appear.
Permit easy reversal of actions	The user should be provided with options to undo actions carried out. As a result, the user loses the fear of committing mistakes and it encourages the exploration of unknown features and applications by the user.
Support internal locus of control	A user should always feel that he has control of the application. This means that the system responds to the action of the user and not vice versa. There should be no unpredictable system actions and the current status should always be easy to query.
Reduce short-term memory load	The short-term memory of a person can not memorize any number of information units. The information should be best displayed on one page and not on multiple pages and windows. In addition, a user will need time to learn complex actions and relationships.

Table 2.9: Golden rules of interface design by Shneiderman [SPCJ13]

Name	Description
Visibility of system status	The system always informs the user about what is happening, timely and with appropriate feedback.
Match between system and the real world	The system speaks the language of the user by using familiar words, phrases and concepts.
User control and freedom	A system should never let users get into situations so that they can not find their way back. It should support undo and redo.
Consistency and standards	Users should not have to think about whether different words, situations and actions mean the same thing. Design standards should be adhered to.
Error prevention	Error-prone situations should be avoided. Moreover, it makes sense to design the system so that it supports users in avoiding mistakes, and if they still occur, offer assistance to solve them.
Recognition rather than recall	The short-term memory of a user is limited. Therefore, you should not overburden them or expect them to remember content. Therefore, all necessary information should be visible or easy to obtain.
Flexibility and efficiency of use	Frequent actions should be user-customizable to allow experienced users to operate faster.
Aesthetic and minimalist design	Dialogues should not contain unnecessary or rarely used information. Every additional piece of information competes with the relevant information and reduces its visibility..
Help users recognize, diagnose, and recover from errors	Good error messages are defensive, precise and constructive. Accurate error messages give the user accurate information about the cause of the problem.
Help and documentation	Sometimes it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Table 2.10: Usability criteria by [Nie94]

2.9 State of the Art

Currently, there exist many usability guidelines for designing web applications, but there is no globally accepted and usable standard. Each of the guidelines is strong in different fields and cannot be interpreted widely, because development processes are taking place under different settings. Styleguides.io is a service providing the most comprehensive list of existing style guides of well-known companies. However, there is no generally applicable style guide for web applications. Currently existing guidelines for web usability are explained in the following.

Standards are published by national or international organizations of standardization. Nielsen et al. [Nie93] differentiated between three big types of standards: national and international, industrial and in-house. International Standardization Organization (ISO), whose standards are largely validated by many years of international research, developed an international standard named "ISO DIS 9241-210"[ISO10], which provides recommendations and guidelines for the user-centered design of web user interfaces. The standard contains also recommendations on data protection and corporate principles for personalization and internationalization. Another standard published is the "ISO 9241-151:2008" [ISOb], which contains recommendations for high-level design decisions and design strategy, content design, navigation, search and content presentation of user interfaces.

In 1990s Nielsen [Nie90] published guidelines, based on extensive usability tests. Nielsen wrote 113 design guidelines in 2001 [Nie01]. Although the web has changed a lot since its introduction, the guidelines are very stable. 80% of the first web usability findings from Nielsen from the 1990 years are still up to date [Nie07]. He identifies the reason for this in that the guidelines are based on human characteristics that do not change much, changes only arise by technical improvements. Horton [Hor05] introduced 87 guidelines, which combine usability with accessibility.

U.S. Department of Health and Human Services (HHS) [Lea06], developed in collaboration with the US General Services Administration 209 guidelines on the basis of numerous publications. Unlike Nielsen also guidelines for the design process and evaluation, for usability testing and accessibility are described here.

The Human Factors International (HFI) Incorporation annually presents guidelines based on research findings in the field of Web usability. These guidelines consider all relevant recommendations and findings including usability methods and dealing with general web design [HFI].

Joint Information Systems Committee for Higher Education (JISC) published 121 web design guidelines as a result of a study, which identified and inspected 200 publications related to usability and HCI for learning, teaching and research [BKS09].

User Focus presented a guideline catalogue with 247 web usability guidelines, which are also available as an Excel worksheet aiming rapid expert-based evaluation [247].

Since the market for mobile devices is booming, mobile usability gained in importance. There are several guidelines dealing with usability of mobile web sites or mobile applications. It is nearly impossible to find structured and evaluated guidelines for the design of mobile applications in the scientific literature, but there are several websites, books and reports concerned with it[SHH+15], which are described as follows:

Budiu and Nielsen [NB11] recognized the lack of guidelines for mobile context and developed 85 guidelines. In their book "Mobile Usability" [NB12] published in 2012, they provide recommendations as a result of usability studies with end users and point out usability problems, which occur in using mobile apps and web sites.

The Mobile Web Best Practices (MWBP) group of W3C published best practices, which, among other things, relate to the usability of mobile websites [RM08]. Passani, critic of these recommendations of MWBP developed 2008 his own guidelines based on best practices [Pas10].

The three biggest market leaders of mobile operating systems Android, iOS and Windows Mobile provide guidelines, tailored to the development of apps for the specific operating system [And, App15, Mic10]. Based on the guidelines of Apple and the MWBP, Griggs et al. [GBG09] developed recommendations for the design of mobile websites.

CHAPTER 3

Methodical Procedure - Case Study

The following description and sequence of the methodical procedure form a retrospectively linear, rational presentation of the design and research work. In reality, however, the design process has been an integrated activity, alternating between producing solutions and investigating the problem.

3.1 Case Study MineralBay

A case study is an improved examination of one or more objects to understand a particular topic and to identify factors that could inspire or influence the researcher. The case study conducted as part of this research focuses on answering the research questions.

This chapter describes the case and refers to the cooperation between the Vienna University of Technology and the Montanuniversität Leoben as part of the project MineralBay. MineralBay is a web-based material management software system for excavated material, whose speciality is access to and processing of real-time data, e.g. in the form of online material analysis results from tunnel boring machines. The information obtained is used for quality management of already placed excavation material and for gapless documentation of the material flow from the beginning to the end of the construction project.

The system is developed for the following parties: clients, contractors, companies and quarries from the raw material industry, landfill operators, construction companies, authorities, etc. As the basic programming language for the project, Java is used for the business logic of the server (the backend) and JavaScript for the user interface (the

front end). The data are administrated using PostgreSQL, which is a relational database management system (RDBMS).

Furthermore the following technologies are used in the MineralBay project:

- Maven: for simplifying the embedding and versioning of external libraries
- Spring: platform offering extensive development functions, which for example simplify the maintenance of the target system
- Hibernate: for simplified and more easily maintained database access
- AngularJS: technology for effective user interface development.
- JHipster: Yeoman-based application generator used to quickly develop web applications using Angular, Bootstrap and the Spring Framework [Pra17].

It also offers interfaces to external systems in order to import information and thus minimise data input. Interfaces to the software of public authorities, geological information systems and project management systems (construction planning) will also be provided [EGG15].

The following domain model shows the main entities of the system and the relationships between them and contains the following central information objects:

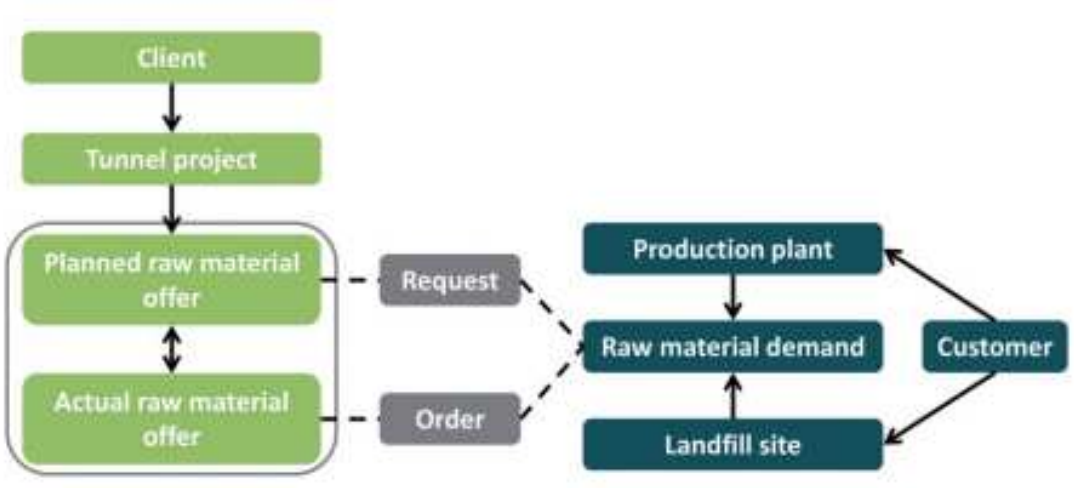


Figure 3.1: Domain model and relationships of MineralBay [EGG15]

Name	Description
Client (material owner)	A public or private client identified through their key data. A client can create and administer several underground construction projects.
Tunnel project	An underground construction project, which is in the design phase or under construction, created and implemented by a client with the entering of additional project data (location, tunnelling method, construction period, geology etc.). A tunnel project produces excavated material from various tunnel sections, which represents the planned raw materials offer.
Planned raw material offer	Technical, chemical und mineralogical specification of the material expected to be excavated from a tunnel section
Actual raw material offer	The actual raw material offer is determined from the data in real time to the technical, chemical und mineralogical specifications of the rock material currently being excavated during a tunnel drive.
Customer (industrial company or landfill operator)	Mineral raw material processing industrial company or landfill operator, which can administrate one or more production/landfill locations
Production location or landfill site	Production location, which wishes to process the excavated rock material or landfill site, which has landfill volume available for excavated rock material.
Raw materials demand	Technical, chemical und mineralogical specification of the raw material needed by an individual production location or for tipping at landfill sites.
Raw material enquiry	If a planned raw material offer matches a planned raw material demand, a customer can inform the client/material owner of his interest in a particular raw material.
Raw material order	If an actual raw material offer matches an actual raw material demand, a customer can order the offered raw material from the client/material owner or alter his raw materials enquiry into an order.

Table 3.1: Information objects of the domain model

3.1.1 Project management

MineralBay uses scrum for the management of the software development process. Scrum is an agile framework, which divides the development into small sub-projects, which are successively implemented in iterations called "sprints". Sprints usually last two to four weeks with the goal to deliver functional, tested and high-quality code. Scrum is based on the experience that the entire development process is unpredictable. For this reason, intermediate results are created. Based on these intermediate results, the missing requirements and solution techniques can be identified efficiently. The long-term plan called "product backlog" is continuously refined and improved. The detailed plan "sprint backlog" is created only for the next sprint.

There are three central roles within the scrum framework:

- Product Owner: acts as the customer or a representative of the end user and is the sole person responsible for managing the product backlog.
- Development Team: is responsible for delivering the product functionality in the order desired by the product owner.
- Scrum Master: is responsible for ensuring that the scrum process works correctly. For this he works together with the development team, but usually does not belong

to it. He introduces the scrum rules, checks their compliance and takes care of the elimination of disturbances and obstacles.

Weekly jour fixes take place where project members come together to compare project status, discuss problem situations and plan next steps.

In order to become familiar with the development process, in which the style guide is intended to be used, I participated in the weekly meetings of the project team. There I had the opportunity to interact with the project team and to observe the current process of the development and the usability design. Beside the attendance at the meetings, I also have been part of the development team for a short time. The development of some tasks for MineralBay made it possible to take the view of the developers and to consider that consistently during the development of the style guide.

3.1.2 User Interface of MineralBay

Since the web application is built using JHipster 2.8.0 the Graphical User Interface (GUI) is based on Bootstrap 3, which makes use of HTML5 Boilerplate. There was no systematic approach for the design of the user interface, thus decisions are made ad hoc by the developers. The only assistance the developers had, were seven wireframes at the beginning of the project (see 8). There were no specifications regarding corporate, user interface and interaction design.

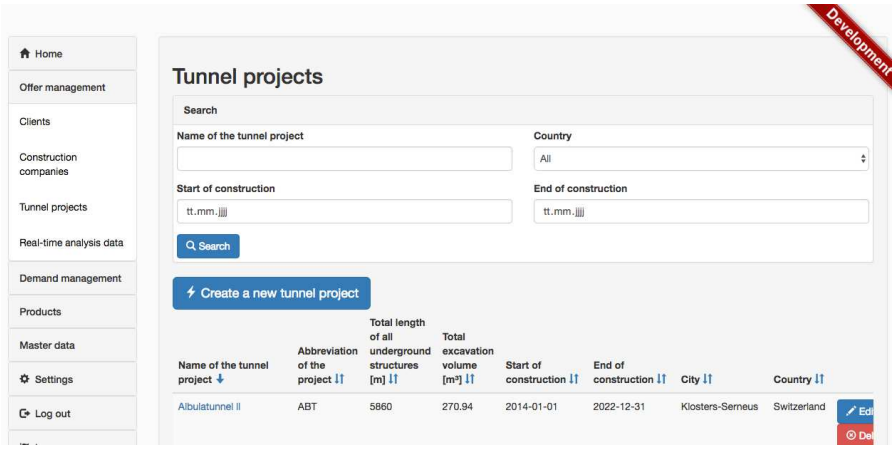


Figure 3.2: Overview of the tunnel projects.

The user interface is divided into two main areas. The area on the left, which contains the side navigation bar, is a direct representation of the domain model 3.1. The area in the middle is used to show the content. As can also be seen in 3.2, the content area is considerably reduced by the page navigation. The user works in a fairly narrow visible area of the content, the full content of which, can be accessed by scrolling horizontally.

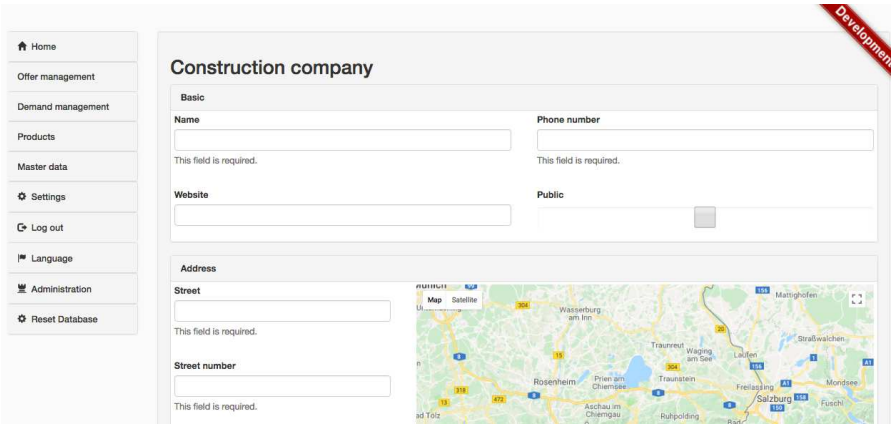


Figure 3.3: Mask for creating a new construction company.

The lack of specification regarding sizes, icons and the distances between different UI elements prevents the content area from being used optimally (see 3.3).

The input of related information of a domain is divided into several pages. The relationships between the entries and the order of the entries are difficult to understand due to the absence of page guidance and the lack of cross-page error messages (see 3.4).

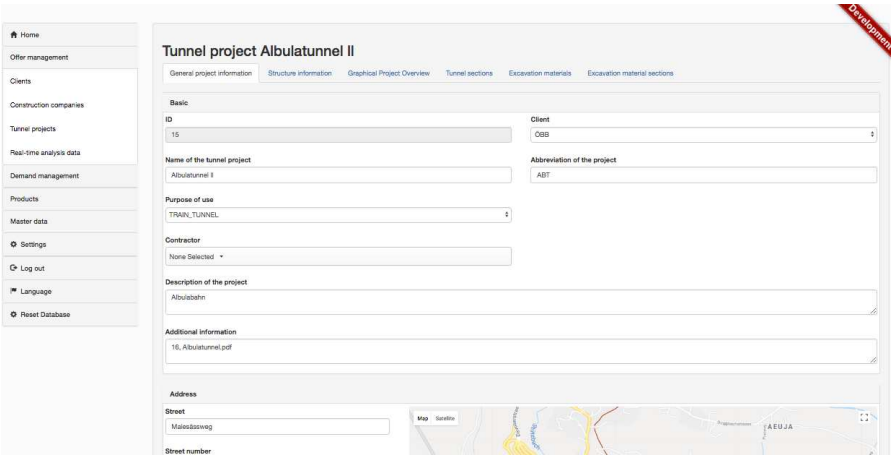


Figure 3.4: Creating and editing of a tunnel project is divided into several pages.

3.2 Methodical Procedure

The procedure of this work is based on the user-centered design process, which has been already presented in chapter 2.3. The phases of the process form the core of the development and do not have a mandatorily defined sequence, so that they can be skipped or repeated at the respective iteration depending on the findings. Depending on the phase

of the development process, there are several different methods for achieving the goal. The methods used for the current research and the significance for further elaboration are presented below. The results are described in the chapters Requirements and User Analysis, Implementation and Results.

3.2.1 Literature Research

First, a literature research was conducted to identify suitable usability engineering methods and to gain knowledge about web usability and style guides. The research area includes the chapters of Fundamentals. This step made it possible to formulate the research question, to guide the work and to define the scope.

3.2.2 User & Requirements Analysis

User and requirement analysis is one of the most important and crucial stages in the development process. The needs and requirements of the users were surveyed in order to design a suitable system and to work preparatory for the implementation of the style guide. Above all, it had to be identified how experienced the target group is in the operation of a computer or a style guide and how often they use them.

3.2.2.1 Questionnaires

A quantitative survey in the form of online questionnaires was used to collect initial information about the potential users and the developers of MineralBay. For this purpose two different questionnaires were created for collecting data from different user groups. The questionnaires mainly consist of closed questions.

In order to introduce MineralBay to the users a presentation video [Oku] has been created using the online web service PowToon [Pow17]. The video has been attached to the questionnaire, so that the users could watch it before answering the questions. The questionnaires were deliberately limited to the most important questions and made available online, so that each participant had to spend as little effort as possible. The time required for both questionnaires was about 5-10 minutes. The participants were notified by mail. In the first few days, a response rate of 43% was recorded. About a week later, a reminder was sent, raising the response rate to 54%. The questionnaires were filled out by 5 developers of MineralBay and 15 students of the Montanuniversität Leoben, who represent the potential users of MineralBay.

In addition to demographic information, the focus was on determining the acceptance of MineralBay, the technical affinity of potential users and their expectations towards MineralBay. On the developer side the previous experiences with style guides and expectations towards the style guide to be developed were identified. To measure the

acceptance the Technology Usage Inventory (TUI) was used [KFH+13]. The TUI model extends the approaches of the previous models such as Technology Acceptance Model (TAM) [Dav85], Technology Acceptance Model 2 (TAM2) [VDC00], Unified Theory of Acceptance and Use of Technology (UTAUT) [VTS+16] and Technology Acceptance Model 3 (TAM3) [VB08] to the two characteristics 'development of cognitive abilities' and 'emotional experiences', which according to Czaja et al. [CCF+06] decisively influence the decision to use a technology. Using the TUI model, the acceptance can be measured using following scales [KFH+13]:

- Interest: Captures a person's basic interest in (new) technologies. The questions do not relate to a particular technology, but ask how much technical knowledge a person has in general and to what extent they keep themselves informed about technological developments.
- Technology Anxiety: Elates the evocation of anxious or emotional reactions through the use of technologies. It records whether a person generally feels overextended by technical devices of all kinds and is afraid of doing something wrong by using the technology.
- Curiosity: This scale captures a person's curiosity about the technology to be evaluated by the TUI.
- Usability: Elevates the perceived ease of use of a particular technology. It records whether a person experiences the technology as user friendly and easy to use.
- Accessibility: This scale captures the perceived accessibility of a particular technology. It represents a person's assessment of whether the technology is financially affordable and easy to obtain.
- Skepticism: Raises the level of mistrust of a person regarding the use of a specific technology. The aim is to determine whether a person views the technology as risky, dangerous and disadvantageous to him.
- Usefulness: Highlights the benefits of a particular technology that a person perceives through their use. It is recorded whether a person considers the technology as useful and whether it believes that the technology can somehow support it in everyday life or facilitate certain activities.
- Immersion: The scale immersion is a special case compared to the other scales, especially since it can only be specified in connection with technologies, which have a virtual world to dip into like television, virtual simulations, computer games, etc.

The immersion scale was not included in the survey, because it can not be applied to the given context. Furthermore, the usability and curiosity were excluded from the survey, since the answering of the questions of these scales is not possible without the finished application. Accessibility was also not part of the survey.

The following tables describe the research goals of the two surveys. The objectives are uniquely identified with the prefix "U" for potential users and "D" for the developers and a sequential number. This identifier has been used as a reference during the evaluation of the questionnaires. The results of the evaluation can be found in chapter 4.

Identifier	Goal
D1	Demographic values of the developer
D2	Degree of knowledge about programming, usability design and style guides
D3	Degree of experience with style guides
D4	Attitude of the developer to the style guide
D5	Content expected from the style guide
D6	Presentation of the content in the style guide

Table 3.2: Research objectives of the survey with the developers

Identifier	Goal
U1	Demographic values of the user
U2	Working environment of the user
U3	Role and responsibility of the user in the work
U4	Degree of experience with the internet
U5	Devices used to access the internet
U6	Features expected from MineralBay

Table 3.3: Research objectives of the survey with the potential users

3.2.2.2 Personas

Based on the findings of the previous steps personas were created to illustrate a potential user and a developer of MineralBay and their expectations towards MineralBay or the style guide. A persona is a concrete, but fictitious person, who represents a typical user. It is important that a persona is as concrete as possible, so that the designers and developers have concrete needs and desires in mind and do not pursue abstract goals.

A focus of the questionnaires was on the characteristics and habits of the users and developers. The gained information formed the personas. After performing the contextual inquiries during the evaluation phase, the persona of a developer had been refined.

The personas were used during the development process and formed the basis for all design decisions. The persona of a potential user is part of the style guide and should

help the developers in understanding them and their needs and thus in designing the user interface.

3.2.3 Design & Implementation

The knowledge gained in the user & requirement analysis form the basis for this phase, aiming to create a uniform foundation for the further development process. At this stage, decisions about the conceptual model were made on an abstract level. To make this visible, prototypes were used. These prototypes were subsequently evaluated and revised with the project management and the supervision. This process was repeated until the desired result was achieved, comparable to the phases four and five of the user-centered design process.

3.2.3.1 Prototyping

Low-fidelity prototyping was used to visualize the findings. For the wireframes of improved look and feel of the application user interface prototyping was used.

3.2.3.2 Style guide

A style guide is a key artifact in Mayhew's Usability Engineering Lifecycle [May99]. It is used to keep the application consistent. Based on the findings from the literature research and the findings of the user & requirement analysis, an interactive style guide has been implemented. The design decisions were made available in the style guide. As described in chapter 2.6.5, the prototypes were split into individual components relevant to the style guide and added to the component library. This step has been repeated whenever the design had changed.

3.2.4 Evaluation

At this stage, the results of the previous phases were evaluated. For this purpose, the qualitative research method "contextual inquiry" was used. The results of the contextual inquires are presented in chapter 6 Results.

3.2.4.1 Contextual Inquiry

In order to evaluate the implemented style guide, contextual inquires were conducted aiming to determine design inconsistencies and usability problem areas within the user interface and to identify lack of needed information regarding the content. The developers of MineralBay were contacted by mail and asked to participate. Dates were agreed with

three developers. In addition to handwritten notes, video recordings were made during the sessions in order to better understand workflows and answers in the evaluation phase.

CHAPTER 4

User and Requirements Analysis

This chapter describes the results of the User & Requirements Analysis described in chapter 3.2.2. The aim of the analysis was to assess the needs and requirements of the potential users, in order to make a design solution for the user interface of MineralBay. Another focus of the analysis was on understanding the developers. Based on the designs for MineralBay a style guide tailored to the needs of the developers has been implemented. The following figure shows the affected phases of the user-centered design process. With this approach, the users and the developers are moved to the center of the design process.

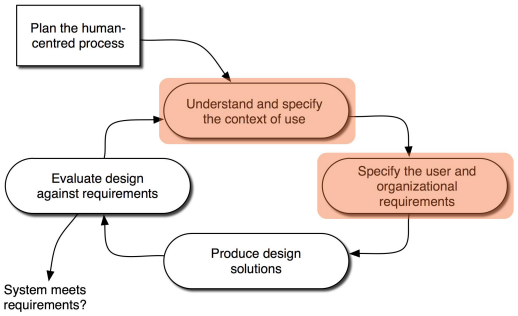


Figure 4.1: User and requirement analysis in context of the user-centered design process

4.1 Evaluation of the questionnaires

For the evaluation of the answers, the statistics and graphics provided by Google Forms were used. In order to evaluate the open questions, the answers were divided into subject

areas. These were used to identify and summarize interrelated passages in the participants' responses to derive consistent or divergent opinions.

The evaluation primarily focused on the goals set for the respective questionnaire [3.2, 6.1]. The objectives addressed with an individual question are illustrated by a reference in square brackets.

4.1.1 Evaluation of the answers of the potential users

In this section, the answers to the survey sent to the potential users of MineralBay are evaluated and also interpreted. Firstly, the sex and age distribution of the subjects is presented. The following image shows, that 11 men and 4 women participated in the survey [U1].

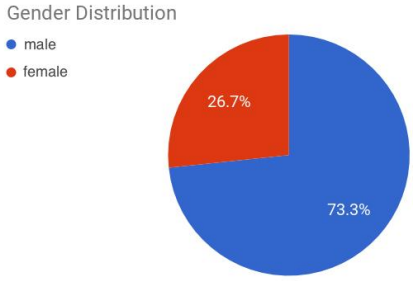


Figure 4.2: Distribution of the gender

Looking at the age distribution, the largest group (46.7%) is between the ages of 20 and 30, followed closely (40%) by the 31 to 40 year olds. The rest of the potential users belong to the age group of 41-60 (see 4.3) [U1] .

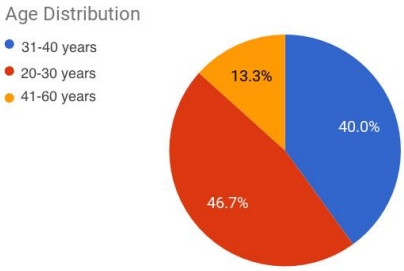


Figure 4.3: Distribution of the age

Almost half of respondents (46.7%) work in an engineering office, followed by those working in a construction company (33.3%). 13.3% of respondents work for a contracting authority. Only 6.7% are self-employed [U2].

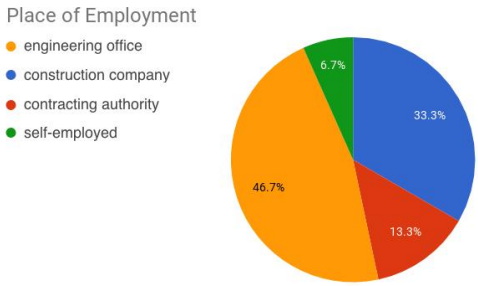


Figure 4.4: Place of Employment

In response to the question about the area of responsibility, most of the potential users (30.8%) answered with production. The rest of the answers were evenly distributed among the remaining selectable areas of responsibility (see 4.5) [U3].

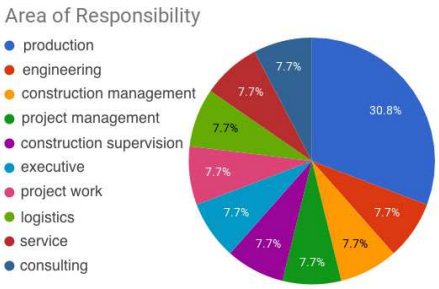


Figure 4.5: Responsibility

All respondents have a university degree [U1]. The result to the question "Since when do you use the Internet", shows that the participants use the Internet for at least 14 years, where the most experienced user has 33 years of experience [U4].

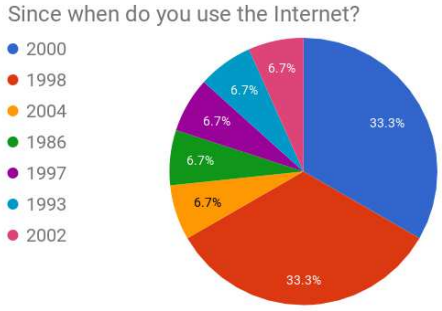


Figure 4.6: Internet usage by the potential users

As a response to the question how many hours in a week the participants spend on the Internet, nearly half (46.7%) of them stated spending between 6 to 15 hours a week, whereas 20% spend 16 to 25 hours and 13.3% up to 35 hours. Only 6.7% stated to spend up to 45, 55 or 70 hours a week [U4].

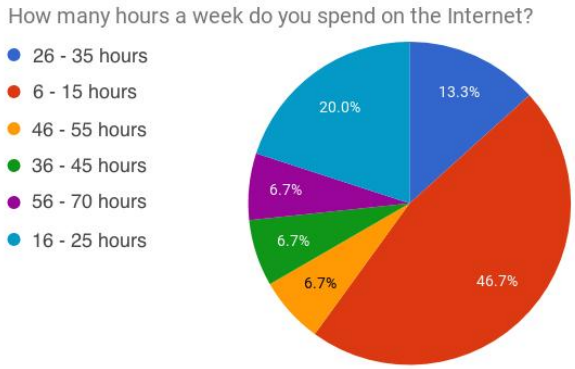


Figure 4.7: Amount of hours spent on the internet in a week

The participants spend most of their time on the Internet at work (93.3%) or at home (73.3%) [U4].

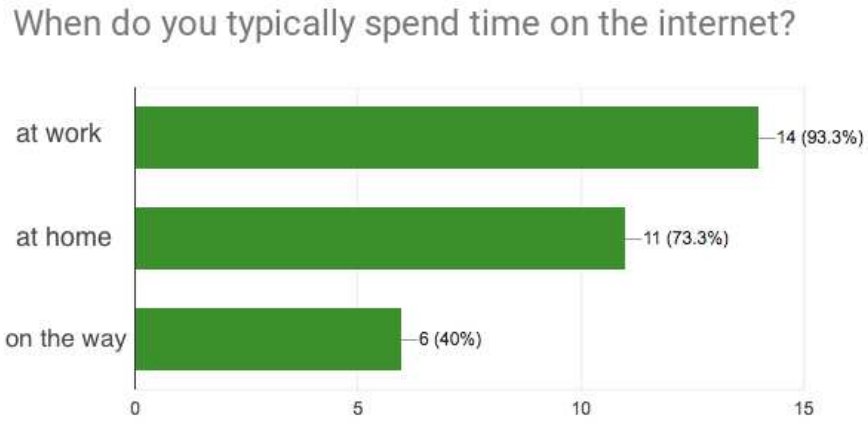


Figure 4.8: Places where the internet is used.

The most common browser used is Mozilla Firefox (66.7%) followed by Internet Explorer (33.3%), Google Chrome (20%), Opera (6.7%) and Microsoft Edge(6.7%) [U5].

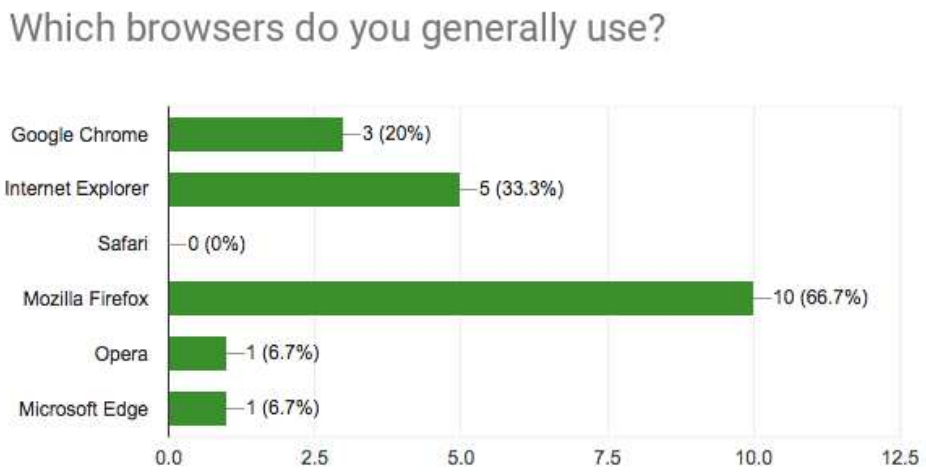


Figure 4.9: Browsers used to access the Internet

As an answer to the question about the devices that users use to access the Internet, 80% stated to use a smartphone in addition to a PC or laptop, which is invariably used by all users, whereas tablets are used by 26.7% [U5] (see 4.10).

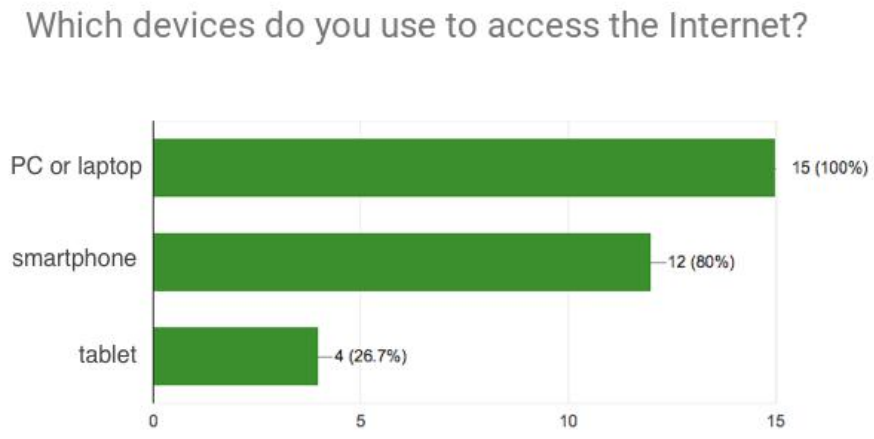


Figure 4.10: Devices used by the potential users to access the Web

The preferences regarding the devices used for the performance of work coincides with

those that would be used to access MineralBay. PCs or laptops are still favored by all of the participants, while tablets are not. Smartphones are/would be used by only 26.7% [U5] (see 4.11).

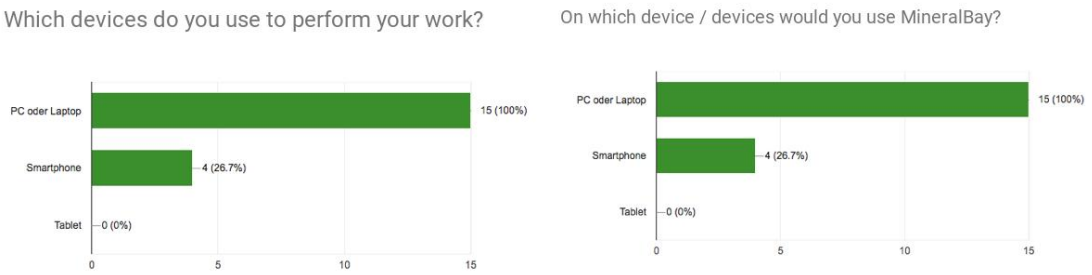


Figure 4.11: Left:Used devices for performing work. Right: Preferred devices to access MineralBay

In the following, further results are described based on the scales interest, technology anxiety, usefulness and skepticism of the TUI model. The calculated mean values of the scales are interpreted on the basis of 1 for "applies" to 5 for "does not apply". The interpretation of the scale interest shows that with a mean of 4.26 the potential users of MineralBay have a great interest in new technologies. The calculated means of 2.96 for technology anxiety and 2.76 for skepticism show that despite the great interest they are not completely open to new technologies, have their fears and that they are also a bit skeptical towards MineralBay, since they see it slightly as risky and dangerous. Nevertheless looking at the scale of usefulness, one can see that almost all respondents (mean 4.9) are convinced of the usefulness of MineralBay.

In addition to the answers to the closed questions, the comments of the participants also gave valuable hints about the expected features from MineralBay, which are summarized as follows [U6]:

- The users want a simple operation, a good overview of the information to be displayed and a search function.
- The application should be free of charge and highlight current or urgent issues. In addition, it should have a notification function.
- The application should show, where materials can be deposited and where aggregates can be purchased cheaply.
- The qualitative and quantitative characteristics of an excavated material, as well as possible types of use and possible customers or suppliers should be provided.
- The users expect a real-time analysis of the material to avoid intermediate placement for material sampling.

4.1.2 Evaluation of the answers of the developers

This section shows the results of the survey sent to the developers of MineralBay. The questionnaire was filled out by 5 developers. Looking at the gender of the participants, it can be seen, that also here the men are in the majority[D1]. All participants belong to the age group 20 - 30 years[D1].

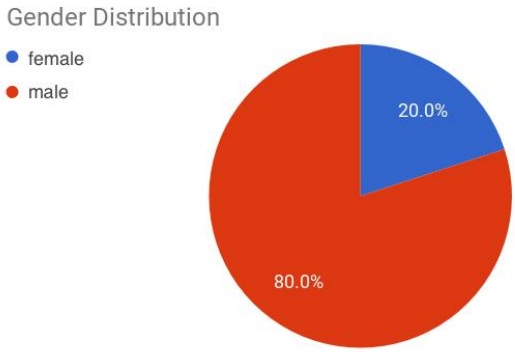


Figure 4.12: Distribution of the gender

The developers have an average programming experience of 6.2 years, whereby the individual experience of developers varies between 3 and 9 years [D2].

The developers were asked to rate their general knowledge in software engineering, usability design and about style guides on a scale of 1 (no knowledge) to 7 (excellent). The majority of the developers (80%) rated their knowlege in software engineering as good and 20% as very good [D2].

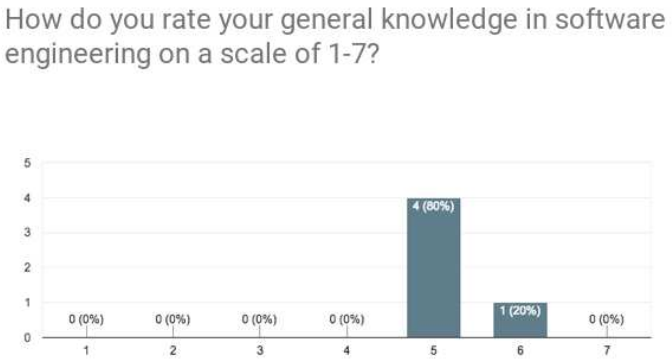


Figure 4.13: Knowledge in Software Engineering

The estimations of the knowledge in usability design were in the lower ranges of the scale [D2]. Thus 40% rated their knowledge as moderate, another 40% as little and the rest as very little.

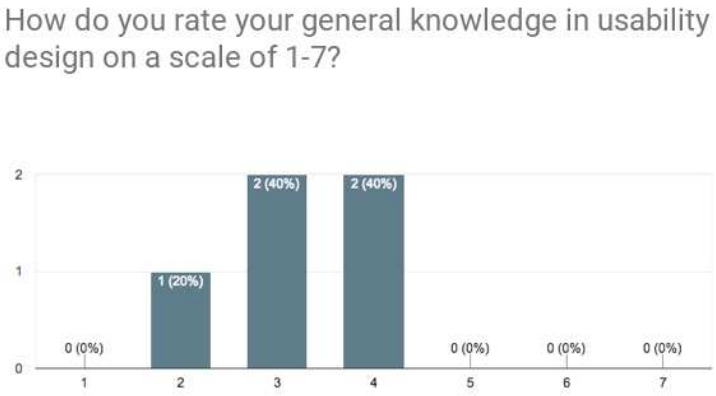


Figure 4.14: Knowledge in Usability Design

The developers have the least knowledge about style guides [D2]. 60% stated to have very little knowledge about style guides and 20% even do not know anything about style guides whereas the remaining 20% have moderate knowledge. As an answer to the question how often they use style guides, 40% chose never, 40% rarely and the rest sometimes [D3]. All developers without an exception see a need for a style guide [D4].

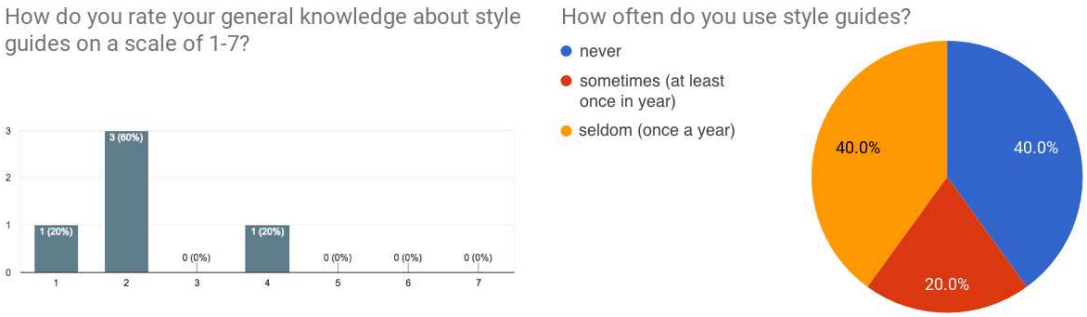


Figure 4.15: Left: Knowledge about Style Guides - Right:Use of Style Guides

In the following, the wishes and suggestions of the developers regarding the style guide, which are commented in the questionnaire, are summed up [D5, D6].

- The developers expect a guide on how to build html pages so that the whole product has a consistent structure and the same look & feel.

- They want design specifications for common UI elements and easy-to-understand instructions on when to use them, so that they have to think as little as possible.
- The style guide should provide a structuring of the tabs, uniform error messages and date pickers, give recommendations on placement of input fields and pattern, which can be copied to the project.
- The reading or understanding of the style guides should not take much time and difficult concepts or guidelines should be supported by examples.
- The style guide should have a good structure so that it is possible to quickly find necessary content.

4.2 Personas

One focus of the two questionnaires was on the characteristics and mindsets of the potential users and developers of MineralBay. Based on the findings of the two questionnaires fictitious profiles were created. The profiles include the routine, the interests and expectations, the profession and also a representative picture.

4.2.1 Developer Jerome Thomas

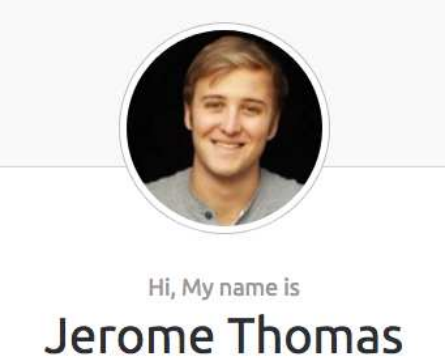


Figure 4.16: Persona of the developer Jerome Thomas

Jerome is 25 years old, studies Computer Science and has more than 3 years of experience in programming. He implements and sometimes also tests the application and has no contact to the end users. Although he rarely uses style guides, he finds them useful. Since Jerome is not an expert in usability design, he prefers to use predefined design decisions. Jerome expects a clear structure, examples with illustrations and code snippets. It is very important for him to easily find what he is looking for.

4.2.2 User Freddie Hopkins

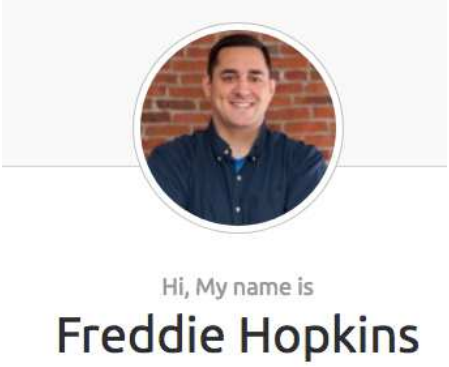


Figure 4.17: Persona of the user Freddie Hopkins

Freddie is 32 years old, studied engineering and works in the construction industry. He has been using the internet for over 17 years and spends about 15 hours a week on it. Freddie finds it more handy to do his work on a PC or a laptop, but in his spare time he also uses a smartphone or a tablet to access the internet. In the course of his life he gained a lot of technical knowledge and is open to new technologies. He sees a need for such an application and thinks that it will support him in his work. Freddie will definitely use the application, if it is free of charge, has a simple operation and a good graphical overview.

CHAPTER 5

Implementation

This chapter gives an insight to the implementation and the build process of the style guide. The wireframes created as part of the design phase of the user-centered design process are presented in the first chapter. According to De La Cuadra [DLC15], the wireframes build the first stage of the style guide driven development process. Thus, they served as a basis for the style guide's component library. After a brief introduction to the technologies used for the development, the final version of the style guide is presented and the improvements based on the results of the contextual inquires are explained.

5.1 Prototypes

5.1.1 Logo Design & Color Scheme

Since there were no branding details and color specifications, a corporate design had to be worked out before beginning the conception of the application. The first drafts of the logo were created using the webservice FreeLogoServices [Inc18] and the raster graphics editor Adobe Photoshop (see 5.1).

An important point was the integration of the designs into the corporate design of the INSO group. An independent design was created, which, however, establishes a relationship to the general corporate identity of the INSO group through targeted color choice and the picking up of typographic elements. To gain an overview of the corporate design of INSO the webservice "Stylify Me" was used. "Stylify Me" allows the user to extract the style guide of a given web site including colours, fonts, sizing and spacing, without the need to inspect each element [YM18]. The extracted color were used, to form the color scheme of MineralBay (see 5.4). Based on the extracted color scheme, four logos were designed (see 5.2).

Figure 5.1: First versions of the MineralBay logo

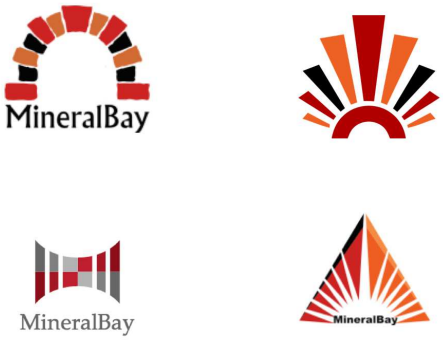


Figure 5.2: Logo suggestions in the colors of the INSO

Finally, the following logo has been found appropriate for MineralBay. The created logo is repeatedly used in the layout of the web application. In the future, this logo can also be applied in other areas, such as presentations, posters and other materials related to MineralBay.

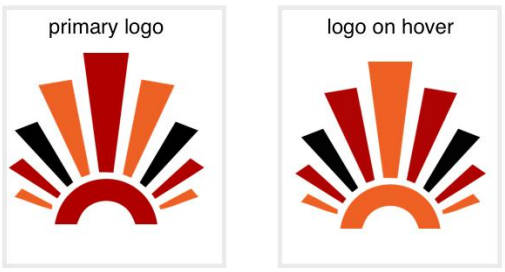


Figure 5.3: Final logo of MineralBay

Figure 5.4: Color scheme of MineralBay

5.1.2 Wireframes of MineralBay

The wireframes were created using the graphics software Balsamiq. The software supports the import of custom images and icons, so the glyphicons used in MineralBay, which are a component of the Bootstrap framework, could be easily imported and used. For the design of the wireframes the information collected through the questionnaires were used with a special attention to the requirements of the INSO team, the representatives of the Montanuniversität and the users, who are represented by the persona Freddie Hopkins (see 4.2.2). The focus during the design of the structure, the layout and the arrangement of the controls was on eliminating the identified problems.

Figure 5.5: Illustration of buttons in MineralBay

The **login page** is inside a frame, which contains the MineralBay logo and floating labels for entering the credentials. The login button has the width of the input fields and is adapted to the color scheme of MineralBay.

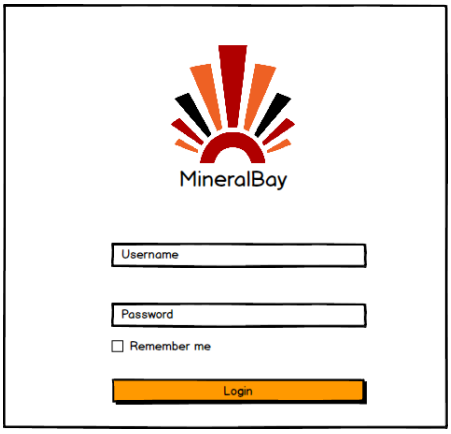


Figure 5.6: Login page of MineralBay

The **navigation bar** in the header replaces the previous side bar. It is divided into static navigation elements, which are located on the top right of an area with the primary red color of the MineralBay theme and dynamic navigation elements, which are placed below. Dynamic menu items vary depending on the role and permissions of the user whereas static navigation items remain the same. The user, who is currently signed in to the system, is shown above the static menu items. The user name is a hyperlink, which navigates to the user settings when clicked. The MineralBay logo is also a link to the start page. By eliminating the side navigation, the content area can use the full width of the screen.

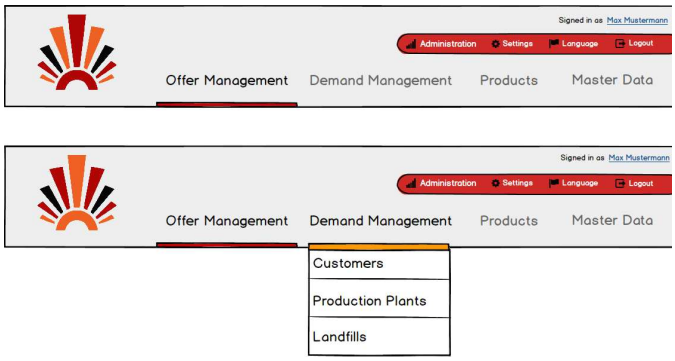


Figure 5.7: Top navigation bar of MineralBay

The **search** forms in the overview pages are placed inside an expandable accordion, which is initially collapsed. The accordion header, as well as the action buttons, are displayed in the main orange color of the MineralBay theme. Figure 5.7 shows an example of the

expanded search form for tunnel projects.

Figure 5.8: Search form for tunnel projects

Another improvement concerns the **location search**. Instead of the previous manual entry of the individual address data such as street, street number, postal code and country, an automatic address completion is introduced. The found address is displayed on the map.

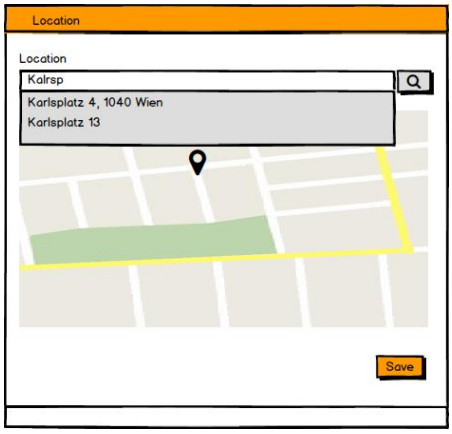


Figure 5.9: Location Search

The **action buttons**, which were in the last column of the tables, are placed inside a dropdown menu. They become visible when the dropdown icon is clicked.

The pages for the creation of different domains that were previously displayed in tabs are now organized with the help of **wizards**, see 5.11. A wizard supports the input of information in a given order, allows the user to keep track of the input progress and to understand relationships between the information entered. By specifying a sequence of inputs and performing intermediate validations, users are less likely to miss important things and will hence make fewer errors.

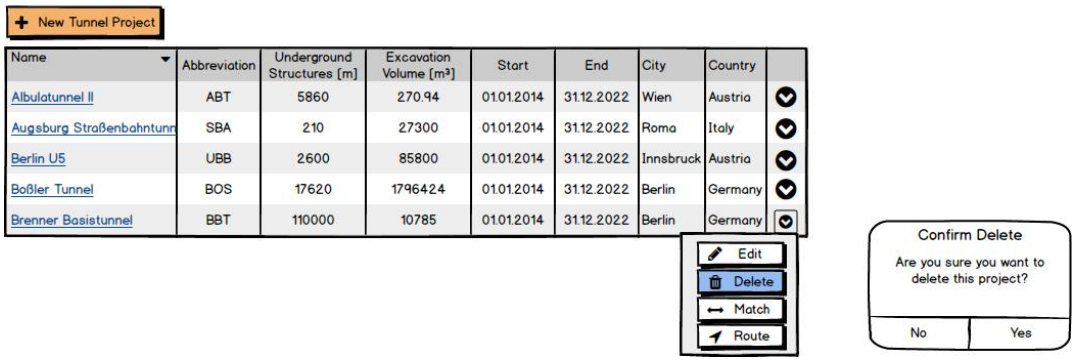


Figure 5.10: Action buttons inside a column

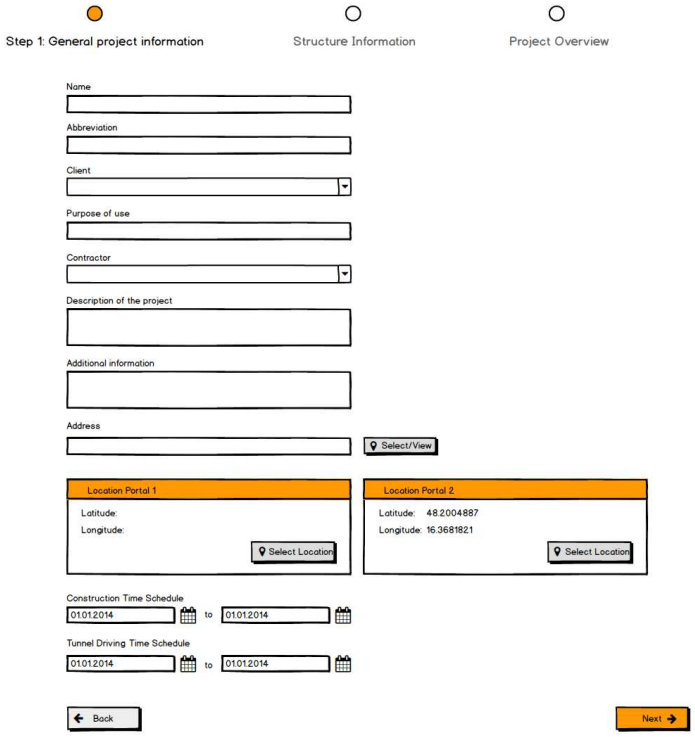


Figure 5.11: Example of a form organized by a wizard

However, the tab navigation is retained to display the stored domain data. The data can be **edited in sections**, see 5.12. By clicking the edit button within a section, the edit mode for this section becomes visible. Thus the user can change the data of this section.

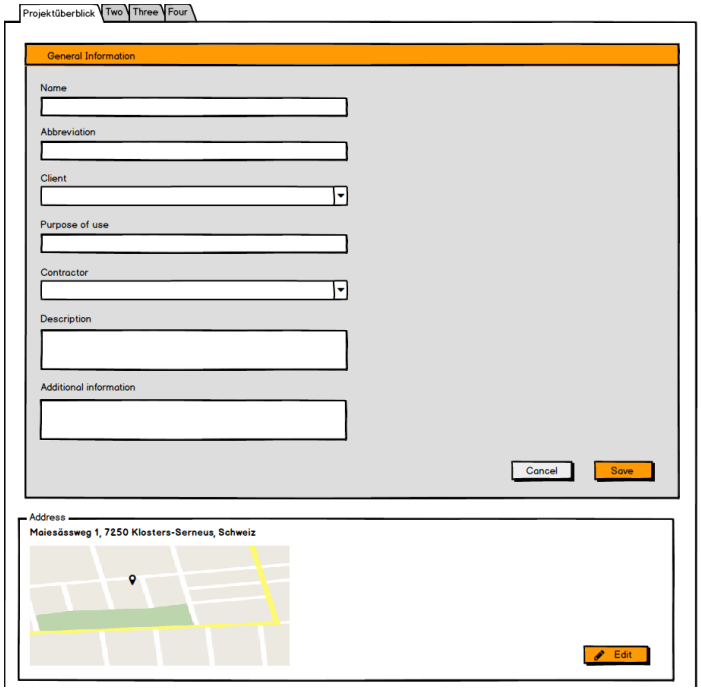


Figure 5.12: Example of a form edited in sections

5.2 Technologies

This section presents the used technologies. The technologies for the realization of the widgets correspond to those in the project MineralBay to ensure a smooth takeover of them. However, different technologies and procedures are used for the build process and for the implementation of the frontend.

5.2.1 AngularJS

AngularJS is a client-side MVC/MVVM Javascript framework that enables the development of web applications with HTML5 & CSS3. Unlike other frontend frameworks, AngularJS uses HTML as a template language, which can be extended with custom HTML tags & attributes.

Model View Controler **(MVC)** is a well-known model used by many programming languages for structuring or architecture in applications and in software development. The MVC pattern divides an application into the following three parts [Goo15]:

- A **model** is the representation of the data of an application, which is usually written in JSON in AngularJS. Beside managing the data, it is also responsible for

all changes. A model can be connected to any number of views. A reuse in other applications is possible without requiring any changes to the logic or data.

- A **view** is the graphical representation of the data, which is the HTML page and the corresponding rendered output in AngularJS. It uses the model to request or save the required data. However, the view is not responsible for the processing of them.
- The **controller** directly accesses the view and the model to make the connection between them and to establish the flow of the data. This is where the business logic resides. It controls the events and is responsible for the evaluation and implementation of the user input and the processing to the related data. For each view, a corresponding controller exists, which handles the changes to the user interface and the forwarding of the data to the model.

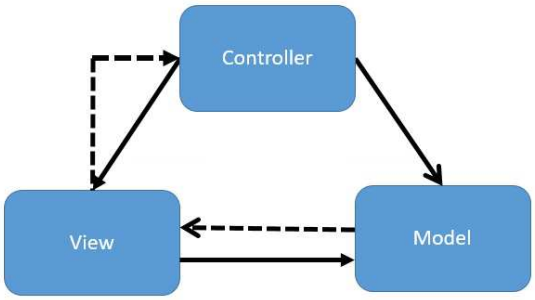


Figure 5.13: Visualisation of the MVC pattern [Beg]

Model View View Model **(MVVM)** is a variant of the MVC pattern for separating layout and program code. Within this pattern, the model takes exactly the same role. However, the view does not need to know its model anymore. The view is linked to the model via the view model, which corresponds to the $scope object in AngularJS. The view model ensures that the view displays the required data in the correct form and is also responsible for automatically transferring new content between the view and the model. This process of data exchange is called data binding.

AngularJS uses neither a pure MVC, nor a pure MVVM implementation, but combines both. It uses the MVVM pattern to provide bidirectional data binding between HTML and JavaScript, but at the same time, it structures the basic design of web applications with MVC. The bidirectional data binding ensures the data to be kept in sync. If a variable is changed in the model, it also changes in the view without the need to trigger an additional event. Additionally, a change of an input field causes an update of the bound variable [Gol, Tod].

In addition to the clean separation of the layers by the appropriate architecture, it is also important to define the folder structure and the distribution of the source code.

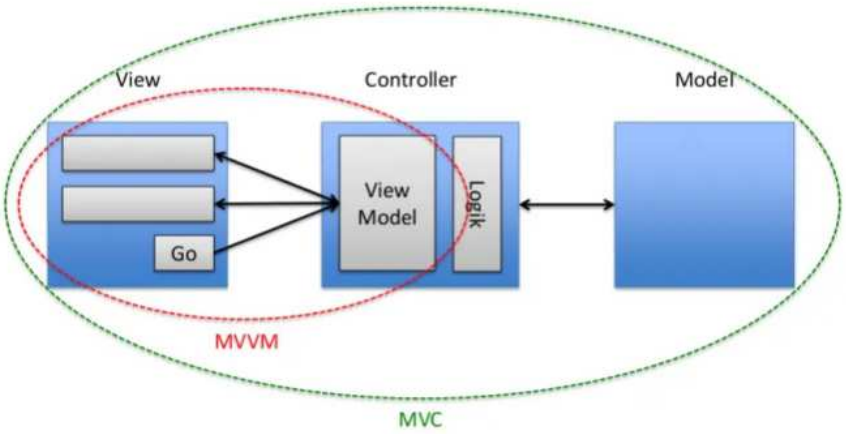


Figure 5.14: Visualisation of the MVVM pattern [Gol]

The structuring of the individual files is the responsibility of the developer. Generally, all JavaScript code can be written in a single file. However, this is not recommended, because the code gets complex very quickly. According to Branas [Bra14b], there are at least 4 ways to organize the code:

- The **inline style** manages an `app.js` file directly next to `index.html`. This variant is only recommended for small projects or just to make a presentation or to evaluate a potential product idea.

- The **stereotyped style** is suitable for small projects with a manageable number of components (controllers, services, directives). All components of a type are developed in a source file. For example, all controllers are included in the controllers.js file.

- The **specific style** is an extension of the stereotypes approach, where each component has its own source file. The files are grouped into folders by the component type. This structure is used when the size of the project increases compared to the stereotype structure.

- The **domain style** is often recommended for large projects. The files are no longer grouped according to their layers or component types, but rather according to domain-specific features. The source files of all component types are located in a folder.

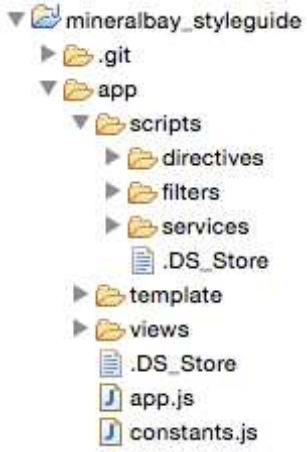


Figure 5.15: File structure of the style guide

The screen 5.15 shows the folder structure of the implemented style guide. It can be seen, that the specific style has been used, since the files are grouped into folders by the component type (directives, filter, services, etc.).

In the following, based on [Tod], the principles and fundamentals of the development with AngularJS are presented.

An AngularJS application is divided into **modules**, which can be defined as follows:

```
app = angular.module('<modulename>', [<dependencies>]);
```

A module can have dependencies on other modules, for which it must have a unique name.

Another important concept of AngularJS is the use of the **`$scope`**. A scope refers to a part of the document object model and represents the view model of the application. Therefore, a scope is an elementary part of the bidirectional data binding. Variables and functions that have to be used in a view must be stored in the same scope. A `$scope` object always refers to a specific controller. Thus an application can have several scopes, which are independent of each other. If data is to be exchanged across controllers, the `$rootScope`, which is available throughout the application, can be used. The `$scope` object has been used in the style guide for the manipulation of all data in the templates. The following snippet from the style guide shows an example of a pager with defined parameters and illustrates the use of scopes.

Listing 5.1: Example of using `$scope` variables in AngularJS

```
$scope.totalItems = 64;
$scope.currentPage = 4;
```

Listing 5.2: The markup representation to listing 5.1

```
<div ng-controller="PagerDemoCtrl">
  <h4>Pager</h4>
  <pre>You are currently on page {{currentPage}}</pre>
  <ul uib-pager total-items="totalItems"
  ng-model="currentPage"/>
</div>
```

Directives extend the vocabulary of HTML. They can be used to create new HTML tags and attributes. In addition to many built-in directives such as ng-app, ng-bind and ng-repeat, it is also possible to define custom directives. This makes it possible to encapsulate entire subsections of an application. The following example shows a directive from the style guide for creating floating labels.

Listing 5.3: Example of using directives in AngularJS

```
app.directive('withFloatingLabel', function () {
  return {
    restrict: 'A',
    link: function ($scope, $element, attrs) {
      var template = '<div class="floating-label">' +
      attrs.placeholder +'</div>'
      $element.after(template);
      $element.removeAttr('placeholder');

      document.querySelector('label[for="'+ attrs.id + '"]')
      .style.display = 'none'

      $scope.$watch(function () {
        if($element.val().toString().length < 1) {
          $element.addClass('empty');
        }
      });
    }
  };
});
```

Listing 5.4: The markup code to listing 5.3

```
<div class="form-group floating-label-wrapper">
  <label for="exampleInputEmail1">Email address</label>
  <input type="email" class="form-control"
  id="exampleInputEmail1" placeholder="Enter email"
  ng-model="user.email" with-floating-label>
</div>
```

A fundamental part of each module are **controllers**. Controllers are construction rules for scopes and define the data and the logic needed for a given view.

```
app.controller('<controllername>', function(<dependencies>)...);
```

They often use services to fill the scope. **Services** provide data across controllers and are also used to exchange information between them. Both are loaded through dependency injection. There are many ways to define a service in AngularJS. A service can be defined using an existing JavaScript object with `.service(object)` or by using the `.factory (...)` function.

5.2.2 NodeJs

Node.js is based on Google's open source JS engine V8 and is a runtime environment and a library that executes JavaScript code outside of a browser. It comes with its own package manager called "Node.js Package Manager" (NPM), which is a command-line tool that allows the installation of packages and the automatic resolution of dependencies to other packages [Tei12]. Packages can be installed with:

```
npm install <package> -save
```

The `-save` parameter makes it possible to save the dependencies during the installation into a file in JSON format named `package.json`. Beside the list of dependencies to other packages the file also contains a definition of the project (name, description and version). When distributing the project, all required dependencies can be installed or updated using the following commands:

```
npm install
npm update
```

Thus, locally stored libraries do not need to be propagated. Figure 5.16 shows an excerpt of the JSON file used for the installation of the necessary libraries of the style guide. NPM has been mainly used for the management of packages that are relevant for the development process like Gulp, etc.

5.2.3 Bower

Just like NPM, bower is another package manager used for the installation and management of packages. However, the types of the available packages differ [Bow]. Registered packages from the NPM repository are primarily intended for execution in the Node.js environment, while bower itself is a package from the NPM repository specializing in managing client-side packages for the browser. Thus, bower has been applied to

```
{
  "name": "mineralbay",
  "version": "1.17.5",
  "devDependencies": {
    "boomsvgloader": "0.0.2",
    "del": "1.2.0",
    "event-stream": "3.3.1",
    "gulp": "^3.9.0",
    "gulp-angular-templatecache": "1.6.0",
    "gulp-autoprefixer": "2.3.1",
    "gulp-bless": "3.0.1",
    "gulp-bower": "0.0.10",
    "gulp-cheerio": "0.6.2",
    "gulp-clean": "0.3.1",
    "gulp-concat": "2.5.2",
    "gulp-connect": "2.2.0",
    "gulp-gh-pages": "0.4.0",
    "gulp-if": "1.2.5",
    "gulp-imagemin": "2.2.1",
    "gulp-insert": "0.4.0",
    "gulp-jshint": "1.11.0",
    "gulp-less": "3.0.3",
    "gulp-livereload": "3.8.0",
    "gulp-markdown": "1.0.0",
    "gulp-mustache": "1.0.2",
    "gulp-newer": "0.5.1",
    "gulp-ngdocs": "0.2.10",
    "gulp-ngmin": "0.3.0",
    "gulp-order": "1.1.1",
    "gulp-plumber": "1.0.1",
    "gulp-rename": "1.2.2",
    "gulp-svg-sprite": "1.2.3",
    "gulp-uglify": "1.2.0",
    "gulp-watch": "4.2.4",
    "highlight.js": "8.6.0",
    "jshint-stylish": "2.0.0",
    "mustache": "2.1.1",
    "vinyl": "0.4.6"
  }
}
```

Figure 5.16: An excerpt of the package.json file of the style guide.

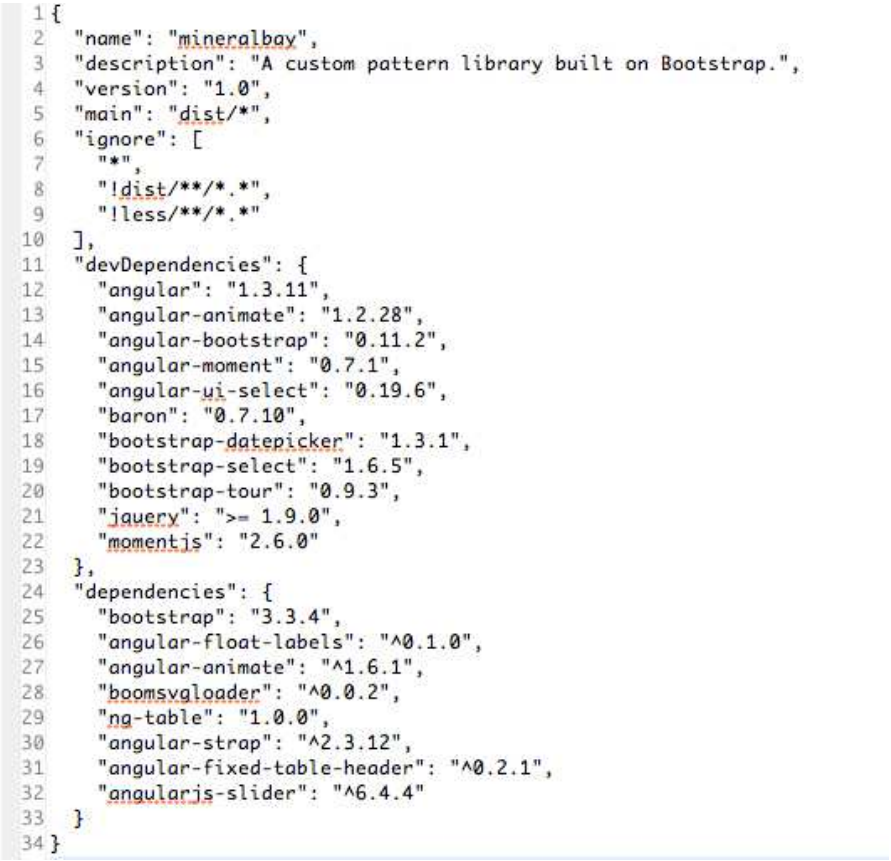
manage the client-side libraries used in the style guide. Again, a configuration file named `bower.json` is used, in which the packages are defined. Figure 5.17 shows the `bower.json` file of the style guide.

Same as for NPM, the –save or –save-dev option on installation causes the package to be added to the dependencies or to the devDependencies array of the configuration file:

```
bower install <package> -save
```

devDependencies are dependencies that are only relevant for the development, but not for the actual execution of the application. Afterwards it is possible to install or update them by using the console commands:

```
bower install
bower update
```



```
{
  "name": "mineralbay",
  "description": "A custom pattern library built on Bootstrap.",
  "version": "1.0",
  "main": "dist/*",
  "ignore": [
    "*",
    "!dist/**/*.*",
    "!less/**/*.*"
  ],
  "devDependencies": {
    "angular": "1.3.11",
    "angular-animate": "1.2.28",
    "angular-bootstrap": "0.11.2",
    "angular-moment": "0.7.1",
    "angular-ui-select": "0.19.6",
    "baron": "0.7.10",
    "bootstrap-datepicker": "1.3.1",
    "bootstrap-select": "1.6.5",
    "bootstrap-tour": "0.9.3",
    "jquery": ">= 1.9.0",
    "momentjs": "2.6.0"
  },
  "dependencies": {
    "bootstrap": "3.3.4",
    "angular-float-labels": "^0.1.0",
    "angular-animate": "^1.6.1",
    "boomsvgloader": "^0.0.2",
    "ng-table": "1.0.0",
    "angular-strap": "^2.3.12",
    "angular-fixed-table-header": "^0.2.1",
    "angularjs-slider": "^6.4.4"
  }
}
```

Figure 5.17: An excerpt of the bower.json file of the style guide.

Bower takes over the installation of the packages and automatically inserts the path to the JavaScript file of the respective package in `index.html`. It is only necessary to add the name of the installed module in the module list of the web app (see 5.5).

Listing 5.5: The current module list of the style guide

```
angular.module('mineralbay', [
    'ui.bootstrap',
    'ui.select',
    'angularMoment',
    'ngSanitize',
    'ngAnimate',
    'ngTable',
    'ngRoute',
    'rzModule',
    'mgcrea.ngStrap',
    'fixed.table.header',
    'mgo-angular-wizard'
```

```
])
```

5.2.4 Gulp

Gulp.js is a task runner based on node.js and offers the possibility to automatically solve regularly occurring tasks in the JavaScript context [Gul]. The Javascript tasks are defined directly in a configuration file named `gulpfile.js` and can be combined with other tasks.

Gulp has been used for the automation of deployments and for live reloading during the development. For automating the deployment, the tasks defined in the gulpfile were executed, e.g. merging of the angular scripts and stylesheets with subsequent compression and copying the files into the dist folder. At the end, the dist folder contains the entire application, which is ready for delivery.

For many use cases there are ready-made plugins, that can be installed directly via NPM. Afterwards, they only have to be configured in the gulpfile for the respective project. One such plugin is "gulp-watch", which has been used in the current project to monitor changes to files and folders and to compile CSS from LESS and to inject the necessary script tags into the HTML head as soon as a new angular module is added and thus to ensure live reloading. The code snippet 5.6 shows a gulp task from the style guide, which ensures the live reloading of the web application by using watch tasks.

Listing 5.6: Example gulp task from the gulpfile of the style guide

```
// Run a server with a watch with gulp server
gulp.task(Tasks.DevelopmentServer, [Tasks.MineralBay],
function() {
  connect.server({
    hostname: 'localhost',
    port: 9000,
    root: 'docs',
    keepalive: false,
    livereload: true
  });

  // Watch Less files
  gulp.watch(['less/**/*.less'],
  [Tasks.MineralBayStylesDev, Tasks.ReloadDevelopmentStyles]);

  // Watch SVG Icon files
  gulp.watch(['svg/**/*.svg'],
  [Tasks.MineralBaySvgIcons, Tasks.ReloadDevelopmentSvgIcons]);

  // Watch Javascript Files and Templates
```

```
  gulp.watch([
    'bower_components/**/*.js',
    'js/**/*.js',
    'app/**/*.js',
    'app/template/**/*.html'
  ], [Tasks.MineralBayJavascript, Tasks.ReloadDevelopmentJS]);

  // Watch html files
  gulp.watch(
    ['app/views/*.html', 'views/**/*.html'],
    [Tasks.CreateDocumentationHTML, Tasks.ReloadDevelopmentHTML]
  );
});
```

5.2.5 Bootstrap

Bootstrap is one of the technologies that have been taken over from MineralBay. Twitter Bootstrap is a CSS and JavaScript framework for designing web pages and provides simple templates to create a portable design. This makes it possible to develop responsive applications, that can also be optimally displayed on different end devices. In addition to many static elements for use in web applications, Bootstrap also offers a variety of interactive features and animations [BO17].

For the widgets to be used in MineralBay, it was necessary to decouple the Bootstrap components from their previous jQuery-based JavaScript and integrate them into AngularJS. There is already a selection of such directives from different providers. Those that are suitable for MineralBay and easy to integrate into the project have been added to the style guide. For this purpose, widgets from several providers such as AngularUI [Anga], UXSolution [UXS], Angular-wizard [Angb], etc. were tested and a for MineralBay suitable AngularJS widget collection of Bootstrap components was created.

5.2.6 Less

LESS is a dynamic language for stylesheets and extends CSS with dynamic behavior such as variables, mixins, calculations, and functions [Les]. Mixins are similar to variables, only extended to CSS classes. A previously written class with arbitrary properties can be included in another class, so it is not necessary to redefine the properties.

Bootstrap is built with LESS, which means that mixins for browser-specific instructions already exist and can be used directly. The bootstrap-specific LESS stylesheets were integrated into the style guide via Bower and were overwritten component by component to adapt the styling of the widgets to the style guide. During the build process, the style sheets are compiled and compressed using Gulp to a common CSS file. Compression

removes all extraneous spaces, line breaks, and semicolons at the last statement in a block to minimize file size. This process is referred to as "minification" and is also applied to JavaScript files. The code snippet 5.7 shows an example of a mixin used for the styling of the tabs in the style guide.

Listing 5.7: Example mixin from the style guide

```
//Overrides Boostrap .tab-focus mixin
.tab-focus() {
  outline: thin dotted;
  outline: 2px auto -webkit-focus-ring-color;
  outline-offset: -3px;
}
```

5.3 A Style Guide for MineralBay

The style guide described below represents the defined design framework for the application MineralBay. Since MineralBay uses Bootstrap, the design of the style guide is based on the principles of Bootstrap, especially in the basic structure and the typical elements. By using the Pattern Library Boomstrap [Boo] as a skeleton code for the development and a template for the design, the used Bootstrap components, the basic layout as well as the classification of the categories, which are strongly aligned to Bootstrap, could be reused, instead of completely rebuilding the user interface. The style guide consists of various immutable and variable components, which are briefly presented in the following chapters.

5.3.1 Basic layout

The basic layout is divided into the categories header, control and content (see 5.18).

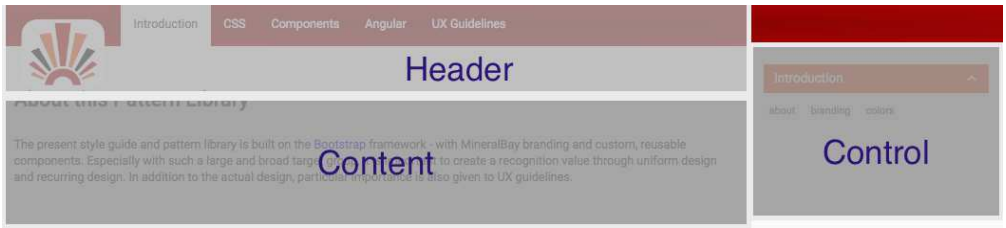


Figure 5.18: Basic layout of the style guide

The header is an immutable component and consists of the MineralBay logo and the menu buttons for the available pages. The control area is a variable component, whose content changes depending on the current shown page. It includes the lateral quick navigation

for orientation and navigation within subpages. Another changeable component is the content area, which reflects the content of the current page.

5.3.2 Logo

The MineralBay logo is available for download in four variations on the introduction page (see 5.19).

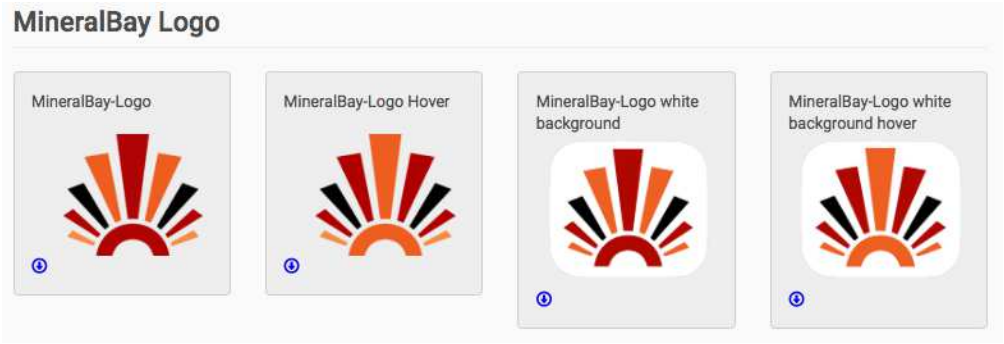


Figure 5.19: Variations of the MineralBay logo

5.3.3 Color scheme

The determination of the colors has already been discussed in the chapter 5.1.1. It should be mentioned here that in addition to the graphical representation of the available colors, the style guide also shows the hexadecimal codes and the respective underlying LESS variable, so that the developer can easily take over this information.

5.3.4 Navigation

The style guide offers two different types of navigations: a main navigation (see 5.20) and a floating side navigation bar (see 5.21) . The menu elements of the main navigation group the content of the style guide into categories, whereas those of the side navigation bar represent links to the content of the current page. In order for the menu items to be recognized as currently selected, they are outlined in color. When hovering over a menu item, they are highlighted in color, allowing the user to see that it is an interactive item.

Figure 5.20: Main navigation of the style guide

The side navigation bar sets an index over the current page and allows the navigation within the current page. It also fulfills a breadcrumb function, since the currently open subpage within the page navigation remains highlighted in color. It also adapts to the changes in the content area, so that the currently visible subpage is always selected in the sidebar. The page navigation always stays at the top of the screen, even when scrolling down. This ensures, that the navigation is always available to use. In addition, a click on the icon in the upper right corner of the side navigation bar, scrolls the page up to the top.

Figure 5.21: Floating side navigation bar of the style guide

5.3.5 Presentation of widgets and codes

All widgets of a category are arranged one below the other and can also be accessed via the links in the side navigation bar. To the right of a widget's name, the official page of the widget is linked (see 1 of 5.22). There are different views for a widget, organized in tabs, at which the currently active tab is highlighted in color. The "preview" tab shows an interactive demonstration of the widget, which allows the developer to test the functionality (see 2 of 5.22).

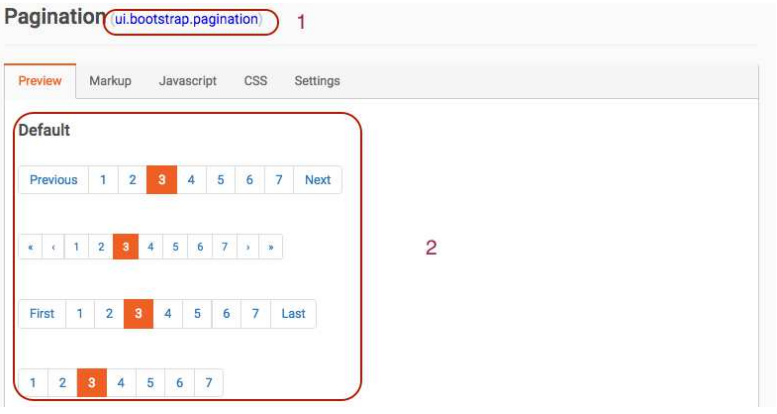


Figure 5.22: Demonstration of a widget

The markup contains the HTML code behind it, whereas JavaScript contains the AngularJS directives or modules (see 5.23). The CSS code, that is necessary for the appropriate styling of the widget is found in "CSS". In addition to the code snippets, a widget may also have a "Settings" and a "Installation" tab. The "Settings" tab provides information about the possible settings of the widget, whereas "Installation" describes the installation process (see 5.24).

Figure 5.23: JavaScript code of a widget

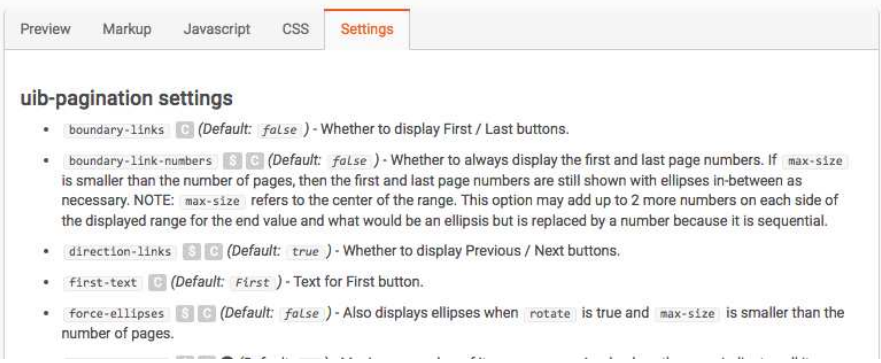


Figure 5.24: Possible settings of a widgets

5.3.6 Responsive design

The style guide and the widgets support the display on small screens such as smartphones or tablets. If the screen gets very small, the page navigation disappears and is placed at the bottom of the page. The menu items of the main navigation also disappear in a submenu, which have to be expanded for use (see 5.25).

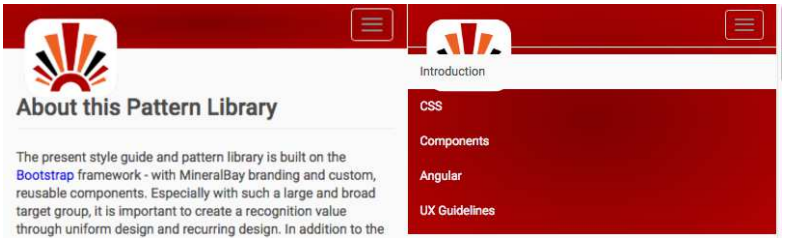


Figure 5.25: Left: menu items of the main navigation disappear; Right: expanded menu items

5.3.7 UX-Guidelines

The style guide also offers educational input in form of guidelines. The guidelines contain best practices for usability design and should be the first place to go for a developer in case of ambiguity. This section is also a good starting point and point of reference for new designers and developers, who can find here at a glance a reference to all available problems and solutions.

Among other things, this section clearly shows which components are to be used in which context and when to choose to use one component over another. For example it identifies the problems associated with using placeholders and suggests floating labels as a good alternative to traditional placeholders (see 5.26).

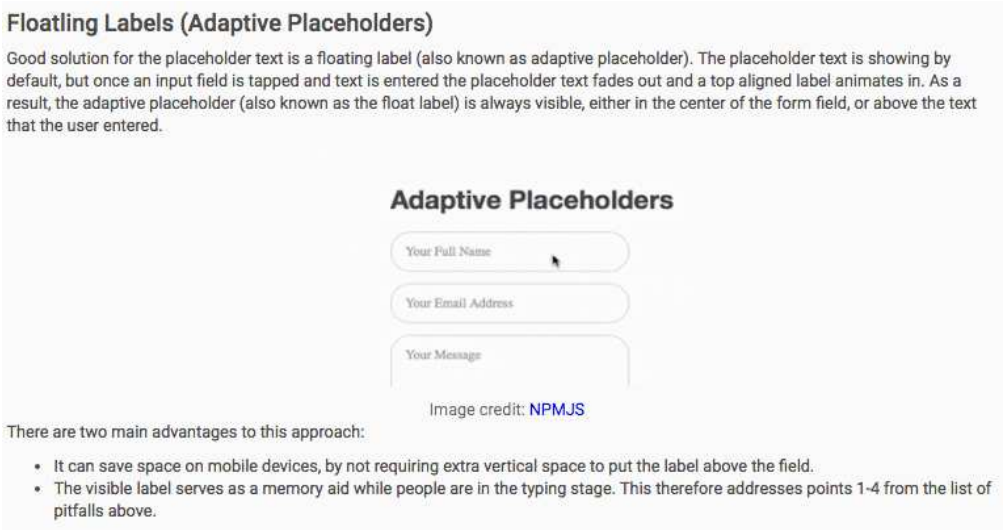


Figure 5.26: Use of floating labels addressed within the UX-Guidelines

It shows common design flaws with real-life examples and the proven solutions to these problems (see 5.27). An already solved problem does not have to be solved again. This

helps to develop solutions more quickly, saving time for further iterations or for dealing with new problems.

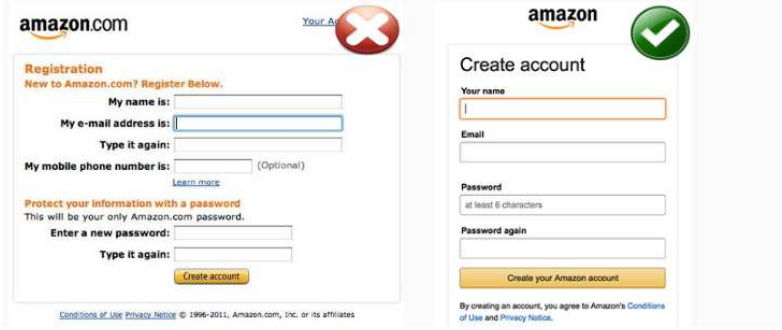


Figure 5.27: Practical tips illustrated with real-life examples

The individual topics addressed are visually appealing through animated gifs and pictures showing do's and don'ts (see 5.28).

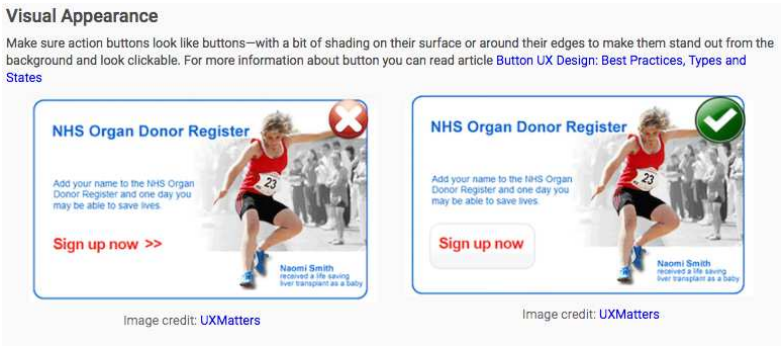


Figure 5.28: An example guideline from the style guide

The guidelines come from a wide collection of different sources. Among others, the Nielsen Norman Group [SHBA+14], known as the world leader in research-based user experience and UXPlanet [Bab].

CHAPTER 6

Evaluation and Results of the Contextual Inquiry

As already described in chapter 4, the development of the style guide was preceded by a user and requirements analysis in order to tailor the style guide to the needs of the developers. This chapter explains how well the implemented style guide meets the needs of the developers. For this purpose, interviews and contextual inquires were carried out. The first part of this chapter discusses the findings from the contextual inquiries. The following section shows further steps taken to eliminate the identified shortages in the style guide.

6.1 Methodology

The observation of the developers, while working with the style guide provided insights about what steps are taken to complete the work tasks. This made it possible to observe differences in the approach of the individual developers as well as basic similarities. All the data collected gives results regarding the weaknesses of the style guide and the behavior expected by the developers.

The developers were initially informed about the purpose of the interviews and contextual inquiries and were very cooperative. Conducting interviews and contextual inquiries provided valuable information about the developer's approach to using the style guide. The two methods complemented each other perfectly. The interviews allowed in-depth questioning on working methods, attitudes and personal opinions, the contextual inquiries provided the equally important context information and allowed to immerse in the work of the developers.

The interviews mainly consisted of open-ended questions. According to Bortz et al. [BD06] open-ended questions promote honest answers from the interviewee, since no answer possibilities are given. As a result, the interviewee can not conclude on the opinion of the interviewer. The questions provided a framework for discussion, which also allowed the statements of the various interviewees to be compared. In addition, some questions had a score sheet, where the developers should assess the relevance using the seven-level scale developed by Likert [Lik32]. The aim was to obtain a clear, measurable statement.

Each session lasted two hours and consisted of the following parts:

Introduction and pre-observation interview

After a small introduction to the procedure and the background for the implementation of the style guide, a small semi-structured interview (see 8) was conducted in order to gain information about the developers and build trust in them. It was emphasized that the inquiry was not about testing the developers's work, but learning from the work process. The developers were asked to agree with the recording of the inquiry and the recording device was set up. This part lasted 10 minutes.

Observation

In the observation part, which lasted about 100 minutes, the developers were asked to solve given tasks (see 8) with the help of the style guide and were asked specific questions about what had been observed, as they went through the work. The tasks were designed to cover all main areas of the style guide. In order to be able to address as many of these tasks as possible, the order of tasks was changed each session. More tasks were planned than necessary, so there would not be any spare time during the contextual inquiries. Attendees were made aware of skipping those tasks where they can not get any further and move on to the next one. The following guideline helped not to lose sight of the focus during the observation:

- Is searched information found quickly?
- Are the links and icons understood correctly?
- How are the different navigation elements understood? Are these intuitive to use?
- Is the structure of the style guide understood correctly?
- Why does the user perform the task? What is his goal?
- How often is the task performed?
- What causes the execution of the task?
- What conditions must be met to complete the task?
- What decisions are made in the execution of the task?

- What information is used for decision-making?
- Is the component to be used recognized correctly?

Post-observation interview

The last part of a contextual inquiry session consisted of an interview, where the gained impressions from the observation part were summed up and specific questions, which did not arise during the observation, were asked. The developers also had the opportunity to ask questions and give feedback to the contextual inquiry. This part lasted about 10 minutes.

6.2 Results

The answers to the questions of the pre and post-observation interviews are summarized below. The questions are identified with the prefix "Q" and the order of the question within the interview.

6.2.1 Results of the pre-observation interviews

Q1: Age of the developers
All three developers where 20-30 years old.

Q2: Programming experience
The developers have an average programming experience of 8.6 years.

Q3: What is your role in the project MineralBay? How long have you been in the project?

Two participants stated that they have been involved in the project for one and two years as developers. A participant is working as a requirement engineer and developer for two years.

Q4: How do you rate your general knowledge in software engineering on a scale of 1-7?
All developers rated their knowlege in software engineering as good(5).

Q5: How do you rate your general knowledge in usability design on a scale of 1-7?
Two of three developers rated their knowledge as little, whereas one of them stated to

have moderate knowledge in usability design.

Q6: How do you rate your general knowledge about style guides or pattern libraries on a scale of 1-7?
The answers to this question are consistent with those of the previous question. Two of three developers rated their knowledge as little, whereas one of them stated to have moderate knowledge regarding style guides or pattern libraries.

Q7: How often do you use style guides / pattern libraries?
One developer rarely uses style guides or pattern libraries whereas the rest stated to use them regularly.

Q8: In which context did you use style guides?
The developers used Bootstrap at work or in university projects or for web development in general.

Q9: What content do you expect from the style guide?
The developers expect that the components as well as colors for buttons are defined and well described. They also want uniform design guidelines.

Q10: How should such a style guide be designed so that you use it in your daily work?
The developers expect the style guide to

- have a simple and clear structure; easy to get around.
- be understandable and have examples with code snippets.
- be like Bootstrap, with a component overview, description and color definitions for messages, buttons, ...

Q11: Do you see a need for a style guide in MineralBay?
All developers without an exception see a need for a style guide.

6.2.2 Results of the post-observation interviews

Q1: What did you like/dislike about the tasks? How hard did you find them? Were they understandable enough?
The developers found the tasks understandable. However, the time was not enough to solve all tasks. A developer stated that the task "Wizard" (see 8) was very complicated

and time-consuming to solve.

Q2: What do you think about the main navigation? How did you find the categorization of it?
The developers found the main navigation to be good and understandable. They especially appreciated the similarity with Bootstrap.

Q3: Did you find the site navigation helpful? What did you like or did you not like so much?
The developers found the side navigation to be useful noting that some categories are quite long (difficult to scroll). Subcategories or bread crumbs would be useful.

Q4: How did you like the pattern detail view or the presentation of the code?
Two of the developers stated that the detail view and the representation of the code is good and clear. A developer would find it better to do not separate markup and preview.

Q5: How did find the performance of the style guide?
All developers found the performance of the style guide good.

Q6: Do you think the style guide can answer your questions regarding usability?
Depends very much on the task, not just on the style guide. In specific terms, e.g: how to center an image, rather not. But otherwise the style guide is very helpful for answering usability related questions. The style guide could not answer when to use a warning or an error. Otherwise, the it could answer all of questions of a developer.

Q7: Are there any missing / obsolete widgets?
The developers found the components to be complete.

Q8: Do you think that the style guide would support you in your development work?
All developers without an exception think that the style guide will support them.

Q9: What did you like about the style guide? What did you not like?
The developers stated that the style guide helps to unify. They appreciated the UX guidelines and different ways of programming a component. Two developers were of the opinion that the style guide provides little information about embedding the CSS code into the project.

6.2.3 Findings

The observations and the answers to the questions were very informative and clearly show, which requirements of the developers are fulfilled and which are not. Overall, the style guide performed well on the planned tasks, which can be found in Appendix A.2. All scheduled sessions were performed and were successful. Only one participant had problems setting up the current development environment at the beginning of the session.

The need for a style guide has been confirmed by all participants. The developers want specifications regarding usability design, which they can follow and believe that the style guide will ensure the uniformity of the user interface. All participants think that the style guide will support them in their development work and would use it regularly. Especially because of the code snippets, which can be copied and pasted easily so that the takeover can be done quickly. The similarity of the main categories to Bootstrap is perceived as familiar and helps the developers to find their way around easily. The separation of AngularJS code and JavaScript code known from Bootstrap is also seen as helpful. The usefulness of the side navigation is also confirmed, as long as the categories do not cover long parts of a page, so that the developers can easily navigate through the content.

The developers were all of the opinion that the components in the component's library and the representation of the code are clear and easy to understand. Especially the previews of the individual components were found to be very useful. In addition the style guide shows different ways of programming a form or other components. The UX guidelines are seen as a good reference, when creating a new form or simplifying complicated input masks. The developers especially appreciate the practical examples.

On the whole, the style guide was able to meet the needs of the developers. However, some suggestions for improvements could be derived, which are described in the following table. The suggestions are identified with the prefix "S" and a sequential number.

6.2.4 Realisation of the suggestions for improvement from the contextual inquiry

In the following, the suggestions for improvement from the contextual inquires and the countermeasures that have been set are summarized.

S1: The introduction should give a brief explanation of the main categories of the style guide.
The introduction page of the style guide has been extended by the description of the main categories.

Identifier	Suggestions for improvement
S1	The introduction should give a brief explanation of the main categories of the style guide.
S2	All widgets should have information about the installation. This can be an installation guide or a link to the widget's page.
S3	Components that occur in multiple places or categories should be linked to each other.
S4	A linking between the components and the associated guidelines is missing.
S5	Some categories of the site navigation are quite long. These can be divided in subcategories or breadcrumbs could be displayed.
S6	A guide on how to change the css code in the project is missing. A description of the difference of CSS and LESS in Bootstrap, would be helpful.
S7	The links to the source sides of the angular components and the UX guidelines were not recognized as such.
S8	A CSS downloadable specifically for MineralBay would be helpful.
S9	The style guide should give information about when to use a warning or an error.

Table 6.1: Research objectives of the survey with the potential users

S2: All widgets should have information about the installation. This can be an installation guide or a link to the widget's page.
The "installation" tab has been supplemented with a reference to the linked page of the widgets.

S3: Components that occur in multiple places or categories should be linked to each other.
Cross references has been added to the components, that occur in multiple categories.

S4: A linking between the components and the associated guidelines is missing.
Widgets for which a guideline applies have been linked to the guideline.

S5: Some categories of the site navigation are quite long. These can be di-

vided in subcategories or breadcrumbs could be displayed
Long categories have been divided into subcategories.

S6: A guide on how to change the css code in the project is missing. A description of the difference of CSS and LESS in Bootstrap, would be helpful.
The introduction page now contains information about the different stylings in bootstrap and which is applied for MineralBay and for the style guide.

S7: The links to the source sites of the angular components and the UX guidelines were not recognized as such.
Since the links are now also referenced inside the "installation" tab, they should be noticeable.

S8: A CSS downloadable specifically for MineralBay would be helpful.
A CSS downloadable would contain the CSS code of all widgets addressed from the style guide. So the downloadable could also contain CSS code of possibly unused widgets. The clarity and maintainability could suffer. This proposal should be reconsidered and has not yet been implemented.

S9: The style guide should give information about when to use a warning or an error.
The style guide already contains this information, which should now be easier to find by the introduction of subcategories.

CHAPTER 7

Discussion

The aim of the current work was the conception and design of the user interface of the application MineralBay and the implementation of an online style guide as an instrument of decision making and information sharing. The focus of the research was on the integration of a style guide to a software development process and the examination of the acceptance of a style guide and expectations towards a style guide. Thus, at the beginning of this work, the following research questions were raised:

- There are many approaches in the literature to integrate a style guide to the software development process. Is the style guide driven development approach suitable for applying a new design to an existing web application?
- What content must be provided, so that a style guide is suitable for developers?

The delineation of the topic to be addressed in the study was challenging. Especially because no preliminary work on usability has been done in the current context. The volume of work was initially underestimated, also because the field of research is large and difficult to define. In addition, the creation of a style guide was associated with a number of different problems and perspectives. In the theoretical part, it would have been possible to focus on the details of the user interface design and to discuss the user interface components in more detail or to deal with the visual perception and ergonomics. These topics were not addressed as they would go beyond the scope of the study.

In this project, due to the history, most of the background information was already available in the project. However, this knowledge was not documented. By attendance in the weekly jour fixes and actively participating in the project as a developer, a lot of knowledge about the project and the requirements could be worked out. In addition, it was possible to take the view of the developers, which was a considerable advantage in

the development of the style guide.

The project has no external client. The fact that there was no customer resulted in a difficulty: getting contacts quickly enough for the analysis of the users. Most of the contacts were students or graduates of the Montan University Leoben, who represented the potential users of MineralBay. For the analysis of the users, questionnaires were used as a method of quantitative research. The students or graduates of the Montan University did not have much time, so that a context-related investigation or an interview was to be excluded. Moreover, the methods probably would not have brought any further information. Due to the continuous interaction with the developers, especially at the beginning of the study, an observational user analysis was combined with the use of questionnaires. The choice of data collection techniques was sufficient. It can be argued that performing the interviews and contextual analysis was definitely worthwhile, as it provided insights into how the developers will work with the style guide, which would not otherwise have been possible. However, the effort was considerable and it is difficult to estimate whether the same results could have been achieved with less effort.

The extensive literature research and the evaluations of the interviews and contextual inquiries provided answers to the questions asked at the beginning of the research.

Thus, a style guide driven development can be defined as method, where the style guide serves as the basis for designing and implementing all of the website's interface. The user interface development is first done in the style guide, after which the implementation is exported to the actual website or application.

For the integration of the style guide to the software development process, the process proposed by De La Cuadra [DLC15] can be applied, which is also described in detail in chapter 2.6.5. According to De La Cuadra the finished design is divided into components to identify the components present in the style guide. Missing components are added to the style guide. These components are used to implement the user interface. The process described has been implicitly applied in the context of the contextual inquiry by the developers to implement the tasks, which actually were the new designs of MineralBay. The developers identified the components of the designs to look for them in the style guide. The found components were combined to implement the design. The style guide served as a starting point for applying the new design to the existing web application MineralBay. Thus, it has been shown that the style guide driven development approach is suitable for applying a new design to an existing web application.

The advantages of a style guide listed in chapter 2.6.2 are an indicator of how important a style guide in a software development process is. The results of the contextual inquires confirm this importance. The developers see a need for a style guide, because of the following reasons:

- The style guide ensures consistency.

- It helps to make design decisions.
- The existence of the components and the possibility of copying the codes speeds up the development process.

The results of the contextual inquires also confirm the recommendations on the content of a style guide, which are described in chapter 2.6.4. However, the developers still saw a need for more content, which are described as follows:

- The introduction should give a brief explanation of the main categories of the style guide.
- All widgets should have information about the installation. This can be an installation guide or a link to the widget's page.
- A guide on how to change code in the project.
- A CSS downloadable containing the styles of all components in the style guide.

The developers especially emphasized the usefulness of the following component-related contents as proposed by Frost [Fro16] and Friedman [Fri13]:

- Code snippets
- Previews of the individual components
- Separation of the code types
- Reference to the source page of the component

In conclusion, the current work is a useful contribution to the research because it takes into account the current knowledge of the subject style guides in general and the applicability of them in software development projects. The research clarifies a broad topic and opens the way for further research, especially with regard to the use and further maintenance of style guides in software development projects.

CHAPTER 8

Conclusion

The present work dealt with the development of an usability concept for the web application MineralBay. The results of the usability research were made accessible via a style guide. The main aim of the diploma thesis was to investigate the applicability of style guides in software development projects. The theoretical framework of the thesis consisted of a detailed literature review of books, articles and lectures by experts working in the field of the design and development of user interfaces. The empirical part provided information about the suitability of the style guide in the current case.

The first part of the thesis dealt with a detailed literature research in the field of user interface design and human computer interaction (HCI). The focus was on the current state of the literature regarding style guides.

As a part of the requirements and user analysis, questionnaires were used as a method of quantitative information gathering. The potential users of MineralBay were asked about their technical affinity and their expectations towards the web application. The developers were also surveyed, in order to find out their expectations towards the style guide and their knowledge in usability design. Based on these findings, a persona of a potential user and developer were created whose goals, motivation and wishes were taken into account during the design process.

The focus of the second part of the work was the conception of a new design for MineralBay and the development of the style guide. Considering the recommendations of De La Cuadra regarding style guide driven development [DLC15], wireframes served as an input for the style guide.

Subsequently, contextual inquires were used as an instrument of qualitative research to find out information about the developers' acceptance of the implemented style guide.

The contextual inquiries were informative and very useful for the research even though they were time-consuming. They provided insights into the day-to-day work of the developers and revealed weaknesses in the style guide. Furthermore, the method helped to understand backgrounds and connections in relation to the work of the developers.

Finally, the results of the contextual inquiries were evaluated and the findings and suggestions of the developers were considered in the style guide.

List of Figures

List of Tables

Acronyms

ACM SIGCHI Special Interest Group on Computer Human Interaction of the Association for Computing Machinery. 5

CHI Computer-Human Interaction. 5

GUI Graphical User Interface. 44

HCI Human-Computer Interaction. 2, 5, 16, 95

HFI Human Factors International. 39

HHS U.S. Department of Health and Human Services. 39

ISO International Standardization Organization. 39

JISC Joint Information Systems Committee for Higher Education. 39

MWBP Mobile Web Best Practices. 40

RUP Rational Unified Process. 17

SE Software Engineering. 16–18

TAM Technology Acceptance Model. 47

TAM2 Technology Acceptance Model 2. 47

TAM3 Technology Acceptance Model 3. 47

TUI Technology Usage Inventory. 47, 56

UCD User-Centered Design. 10, 11, 18

UE Usability Engineering. 13, 16–18, 20

UTAUT Unified Theory of Acceptance and Use of Technology. 47

Bibliography

[247] 247 web usability guidelines.

[AKSS03] Alain Abran, Adel Khelifi, Witold Suryn, and Ahmed Seffah. Usability meanings and interpretations in ISO standards. *Software Quality Journal*, 11(4):325–338, 2003.

[And] Android. Android Design Principles | Android Developers.

[Anga] AngularUI for AngularJS.

[Angb] Easy to use Wizard library for AngularJS.

[App15] Apple. iOS Human Interface Guidelines: Design Principles, 2015.

[AZ05] Boonlit Adipat and Dongsong Zhang. Interface design for mobile applications. In *AMCIS*, 2005.

[Bab] Nick Babich. Designing More Efficient Forms: Structure, Inputs, Labels and Actions.

[BD06] Jürgen Bortz and Nicola Döring. *Forschungsmethoden und Evaluation: für Human und Sozialwissenschaftler*, volume 53. Springer Medizin Verlag, Heidelberg, 4 edition, 2006.

[Beg] Markus Begerow. Model View Controller Pattern Definition & Erklärung | Datenbank Lexikon.

[Ben84] J. Bennett. Managing to meet usability requirements establishing and meeting software development goals. jan 1984.

[Bey99] Karen; Beyer, Hugh; Holtzblatt. Contextual Design. *Interactions*, (January + February):32–42, 1999.

[BGW02] Tom Brinck, Darren Gergle, and Scott D Wood. *Designing web sites that work: Usability for the web*. 2002.

[BKS09] Nigel Bevan, Sara Kincla, and Serco Usability Services. HCI Design Foundation Study. feb 2009.

[BMS05] Michael Burmester, Joachim Machate, and Nina Sandweg. Integration benutzerzentrierter Methoden in die Software-Entwicklung (Integrating User-Centered Design Methods with Software Engineering). *i-com*, 4(3/2005):31–40, jan 2005.

[BO17] Bootstrap and Mark Otto. About Bootstrap, 2017.

[Boo] Introduction | Boomstrap Pattern Library.

[Boo89] P.A. Booth. *Introduction to Human-Computer Interaction*. 1989.

[Bow] Bower — a package manager for the web.

[Bra14a] Brad Frost. Style Guides | Brad Frost, 2014.

[Bra14b] Rodrigo. Branas. *AngularJS essentials : design and construct reusable, maintainable, and modular web applications with AngularJS*. Packt Pub, 2014.

[Car00] John M. Carroll. Making Use: Scenario-Based Design of Human-Computer Interactions. oct 2000.

[Car13] John M. Carroll. Human Computer Interaction - brief intro, 2013.

[CCF+06] Sara J. Czaja, Neil Charness, Arthur D. Fisk, Christopher Hertzog, Sankaran N. Nair, Wendy A. Rogers, and Joseph Sharit. Factors predicting the use of technology: Findings from the Center for Research and Education on Aging and Technology Enhancement (CREATE). *Psychology and Aging*, 2006.

[CL99] Larry L Constantine and Lucy A D Lockwood. *Software for use: a practical guide to the models and methods of usage-centered design*, volume 32. 1999.

[Coy15] Chris Coyier. Where Style Guides Fit Into Process | CSS-Tricks, 2015.

[CRC11] Alan Cooper, Robert Reimann, and David Cronin. *About Face 3*. 2011.

[Dah06] Markus Dahm. *Grundlagen der Mensch-Computer-Interaktion*. 2006.

[Dav85] F D Davis. A technology acceptance model for empirically testing new end-user information systems: Theory and results. *Management*, 1985.

[Deb] Anna Debenham. *FRONT-END STYLE GUIDES : creating and maintaining style guides for websites.*

[DFAB04] A Dix, J Finlay, G D Abowd, and R Beale. *Human-Computer Interaction*, volume Third. 2004.

[DLC15] Adri De La Cuadra. Style Guide Driven Development: A How To Guide To Improve your Development Workflow, 2015.

[DLH02] Douglas K. Van Duyne, James Landay, and Jason I. Hong. The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience. jul 2002.

[DM00] Ruth Dickstein and Vicki Mills. Usability Testing at the University of Arizona Library: How to Let the Users in on the Design. *Information Technology and Libraries*, 19(3):144, sep 2000.

[DP01] Paulo Da Silva and Norman W. Paton. A UML-Based Design Environment for Interactive Applications. *Connect*, pages 60–71, 2001.

[DR99] Joseph S Dumas and Janice C Redish. *A practical guide to usability testing*, volume Rev. ed. 1999.

[DR13] Z Durdik and R H Reussner. On the appropriate rationale for using design patterns and pattern documentation. *QoSA 2013 - Proceedings of the 9th International ACM Sigsoft Conference on the Quality of Software Architectures*, 2013.

[DZN07] Markus Düchting, Dirk Zimmermann, and Karsten Nebe. *Human-Computer Interaction. Interaction Design and Usability*, volume 4550 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, jul 2007.

[EGG15] Hartmut Erben, Robert Galler, and Thomas Grechenig. MineralBay - the portal for raw materials and projects from subsurface construction / MineralBay - das Portal für mineralische Rohstoffe und Projekte aus dem Untertagebau. *Geomechanics and Tunnelling*, 2015.

[Exp11] User Experience. iOS Human Interface Guidelines. *Strategies*, 2011.

[Fa03] Elizabeth Furtado and Et al. Improving usability of an online learning system by means of multimedia, collaboration, and adaptation resources. In *Usability Evaluation of Online Learning Programs*, pages 69–86. 2003.

[Fer03] Xavier Ferré. Integration of Usability Techniques into the Software Development Process. In *Proceedings of ICSE 2003 Workshop on Bridging the Gaps Between Software Engineering and Human-Computer Interaction, May 3-4, 2003, Portland, Oregon, USA*, pages 28–35, jan 2003.

[Fis12] Holger Fischer. Integrating usability engineering in the software development lifecycle based on international standards. In *Proceedings of the 4th ACM SIGCHI symposium on Engineering interactive computing systems - EICS '12*, pages 321–324. ACM Press, 2012.

[Fli08] U Flick. Methoden-Triangulation in der qualitativen Forschung. In *Triangulation. Eine Einführung*. 2008.

[Fri13] Vitaly Friedman. Taking The Pattern Library To The Next Level — Smashing Magazine, 2016-10-13.

[Fro] Brad Frost. *Atomic design*.

[Fro16] Brad Frost. Anatomy of a Pattern in a Pattern Library | Brad Frost, 2016.

[GBG09] Kim Griggs, Laurie Bridges, and Hannah Gascho Rempel. The Code4Lib Journal – library/mobile: Tips on Designing and Developing Mobile Web Sites, 2009.

[Gil09] Alan Woolrych Gilbert Cockton, Darryn Lavery. Inspection based evaluations. In *Human-Computer Interaction: Development process*, pages 273 –276. 2009.

[GL85] John D. Gould and Clayton Lewis. Designing for usability: key principles and what designers think. *Communications of the ACM*, 28(3):300–311, mar 1985.

[Glu97] Myke Gluck. A Descriptive Study of the Usability of Geospatial Metadata :: OCLC Research, 1997.

[Gol] Golo Roden. Webanwendungen mit AngularJS | heise Developer.

[Goo15] Google. AngularJS — Superheroic JavaScript MVW Framework, 2015.

[Gou88] John D. Gould. How to Design Usable Systems, 1988.

[GP05] RN Griffiths and L Pemberton. Don't write guidelines write patterns. *Retrieved August*, 2005.

[Gra] Toni Granollers. User Centred Design Process Model. Integration of Usability Engineering and Software Engineering.

[GS96] Stephen Gale and Stephen. A collaborative approach to developing style guides. In *Proceedings of the SIGCHI conference on Human factors in computing systems common ground - CHI '96*, pages 362–367, New York, New York, USA, 1996. ACM Press.

[Gui95] Ronald A. Guillemette. The evaluation of usability in interactive information systems. pages 207–221, jul 1995.

[Gul] gulp.js - The streaming build system.

[HFI] UI Design Newsletter – Annual Research Review 2006.

[HH93] Deborah Hix and H. Rex Hartson. Developing user interfaces: ensuring usability through product & process. may 1993.

[HH08] Reinhard Höhn and Stephan Höppner. *Das V-Modell XT: Grundlagen, Methodik und Anwendungen*. Springer-Verlag, 2008.

[Hol05] Andreas Holzinger. Usability engineering methods for software developers. *Communications of the ACM*, 48(1):71–74, 2005.

[Hor05] Sarah Horton. Access by Design: A Guide to Universal Usability for Web Designers. aug 2005.

[HR07] Martin Host and Per Runeson. Checklists for Software Engineering Case Study Research. In *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, pages 479–481, 2007.

[HR12] Christian Hauri and Stefania Rosati. Die nachhaltige Einführung und Verankerung von User Experience in Unternehmen. 2012.

[Inc18] LogoMix Inc. Freelogoservices.com, 2018.

[IRRR13] Rodolfo Inostroza, Cristian Rusu, Silvana Roncagliolo, and Virginica Rusu. Usability heuristics for touchscreen-based mobile devices. In *Proceedings of the 2013 Chilean Conference on Human - Computer Interaction - ChileCHI '13*, pages 24–29, New York, New York, USA, nov 2013. ACM Press.

[ISOa] ISO 9241-11:1998 - Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability.

[ISOb] ISO 9241-151:2008 - Ergonomics of human-system interaction – Part 151: Guidance on World Wide Web user interfaces.

[ISO10] ISO. Ergonomics of human-system interaction - Part 210: Human-centred design for interactive systems (ISO 9241-210:2010(E)), 2010.

[JK96] Bonnie E. John and David E. Kieras. The GOMS family of user interface analysis techniques: comparison and contrast. *ACM Transactions on Computer-Human Interaction*, 3(4):320–351, dec 1996.

[JK05] Bill Jerome and Rick Kazman. Surveying the Solitudes: An Investigation into the Relationships between Human Computer Interaction and Software Engineering in Practice. pages 59–70. Springer Netherlands, 2005.

[JLMS03] Natalia Juristo Juzgado, Marta López, Ana María Moreno, and Maria Isabel Sánchez Segura. Improving software usability through architectural patterns. In *Proceedings of ICSE 2003 Workshop on Bridging the Gaps Between Software Engineering and Human-Computer Interaction, May 3-4, 2003, Portland, Oregon, USA*, pages 12–19, jan 2003.

[Jok02] Timo Jokela. A Method-Independent Process Model of User-Centred Design. pages 23–38, aug 2002.

[KFH+13] Oswald D Kothgassner, Anna Felnhofer, Nathalie Hauk, Elisabeth Kastenhofer, Jasmine Gomm, and Ilse Kryspin-Exner. *Technology Usage Inventory (TUI): Manual*. apr 2013.

[Kim02] Kyunghye Kim. A model of digital library information seeking process (dlisp model) as a frame for classifying usability problems. jan 2002.

[KK95] Masaaki Kurosu and Kaori Kashimura. Apparent usability vs. inherent usability. In *Conference companion on Human factors in computing systems - CHI '95*, pages 292–293, New York, New York, USA, may 1995. ACM Press.

[KR05] Mike Kuniavsky and Srinivas Raghavan. Guidelines are a tool: building a design knowledge management system for programmers. In *Proceedings of the 2005 conference on Designing for User eXperience*, 2005.

[Kra13] Dennis Krannich. *Mobile System Design: Herausforderungen, Anforderungen und Lösungsansätze für Design, Implementierung und Usability-Testing Mobiler Systeme.* Books on Demand, 2013.

[KSH+14] Rekha Kengeri, Cheryl D. Seals, Hope D. Harley, Himabindu P. Reddy, and Edward A. Fox. Usability study of digital libraries: ACM, IEEE-CS, NCSTRL, NDLTD. *International Journal on Digital Libraries*, 2(2-3):157–169, mar 2014.

[Lan] Sabine Lang. Empirische Forschungsmethoden Skript zur Lehrveranstaltung. Technical report.

[LAU16] PAGE LAUBHEIMER. Front-End Style-Guides: Definition, Requirements, Component Checklist, 2016.

[Lea06] Michael O Leavitt. *Research-Based Web Design & Usability Guidelines*, volume 1. US General Services Administration, 2006.

[Les] Getting started | Less.js.

[Lew14] Jordan Lewis. Styleguide Driven Development - We build Envato, 2014.

[Lik32] R Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 1932.

[LR94] Clayton Lewis and John Rieman. Task-Centered User Interface Design:A Practical Introduction. 1994.

[Mai] Mailchimp. Welcome to the Mailchimp Content Style Guide | Mailchimp Content Style Guide.

[May99] Deborah J Mayhew. *The usability engineering lifecycle : a practitioner's handbook for user interface design.* 1999.

[MB05] Hans Jürgen Meinhardt and Astrid Beck. Usability im neuen V-Modell XT, 2005.

[McK01] Stuart McKibbin. Designing Web Usability: The practice of simplicity, 2001.

[Met01] Eduard Metzker. Computer Aided Improvement of Human-Centered Design Processes. *Mensch & Computer*, 2001.

[Mic10] Microsoft. Design Guidelines for Windows Mobile, 2010.

[MN95] Robert L. Mack and Jakob Nielsen. Usability inspection methods: executive summary. pages 170–181, jun 1995.

[MNL+93] Robert Mack, Jakob Nielse N, Hewlitt-packard Laboratories, Patricia Brooks, Bell North, and Robin Jeffries Hewlitt-packard Laboratories. *Usability Inspection Methods :*. 1993.

[Mor97] David L. Morgan. *Focus Groups as Qualitative Research*. 1997.

[MR02] Eduard Metzker and Harald Reiterer. Evidence-Based Usability Engineering. In *Computer-Aided Design of User Interfaces III, Proceedings of the Fourth International Conference on Computer-Aided Design of User Interfaces, May, 15-17, 2002, Valenciennes, France*, pages 323–336, jan 2002.

[MRC06] Maristella Matera, Francesca Rizzo, and Giovanni Toffetti Carughi. Web usability: Principles and evaluation methods. In *Web Engineering*, pages 143–180. Springer Berlin Heidelberg, 2006.

[MVP04] Céline Mariage, Jean Vanderdonckt, and Costin Pribeanu. State of the Art of Web Usability Guidelines. In *The Handbook of Human Factors in Web Design*. 2004.

[MVSC05] Ji-Ye Mao, Karel Vredenburg, Paul W. Smith, and Tom Carey. The state of user-centered design practice. *Communications of the ACM*, 48(3):105–109, 2005.

[NB11] Jakob Nielsen and Raluca Budiu. Usability of Mobile Websites, 2011.

[NB12] Jakob Nielsen and Raluca Budiu. *Mobile Usability*. 2012.

[ND86] Donald A Norman and Stephen W Draper. *User Centered System Design; New Perspectives on Human-Computer Interaction*. 1986.

[NeC00] Nuno Jardim Nunes and João Falcão e Cunha. Towards a UML profile for interaction design: the wisdom approach. pages 101–116, oct 2000.

[Nie90] Jakob Nielsen. Designing user interfaces for international use. jun 1990.

[Nie92] J Nielsen. Finding usability problems through heuristic evaluation. *Proceedings of the SIGCHI conference on Human*, pages 373–380, 1992.

[Nie93] Jakob Nielsen. *Usability Engineering*, volume 44 of *Interactive Technologies*. Morgan Kaufmann, 1993.

[Nie94] Jakob Nielsen. Enhancing the explanatory power of usability heuristics. *Conference companion on Human factors in computing systems CHI 94*, pages 152–158, 1994.

[Nie01] Jakob Nielsen. 113 Design Guidelines for Homepage Usability (Jakob Nielsen), 2001.

[Nie03] Jakob Nielsen. Usability 101 : Introduction to Usability. *All Usability*, 9:1–10, 2003.

[Nie07] Jakob Nielsen. Change vs. Stability in Web Usability Guidelines, 2007.

[Nie12] Jakob Nielsen. Usability 101 : Introduction to Usability. *All Usability*, 9:1–10, 2012.

[NP11] Karsten Nebe and Volker Paelke. Key requirements for integrating usability engineering and software engineering. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6761 LNCS:114–120, 2011.

[NZP08] Karsten Nebe, Dirk Zimmermann, and Volker Paelke. Integrating Software Engineering and Usability Engineering. In Shane Pinder, editor, *Advances in Human Computer Interaction*, volume 1997, chapter 20, pages 331–350. In-Teh, 2008.

[OB08] Antti Oulasvirta and Stephen Brewster. Mobile human-computer interaction. *International Journal of Human Computer Studies*, 66(12):833–837, 2008.

[OBW01] Richard Oed, Anja Becker, and Elke Wetzenstein. Welche Unterstützung wünschen Softwareentwickler beim Entwurf von Bedienoberflächen? pages 355–364. Vieweg+Teubner Verlag, 2001.

[Oku] Hafize Okunakol. Mineralbay deutsch.

[OP02] Alexei Oulanov and Edmund J.Y. Pajarillo. CUNY+ Web: usability study of the Web-based GUI version of the bibliographic database of the City University of New York (CUNY). *The Electronic Library*, 20(6):481–487, dec 2002.

[Pas10] Luca Passani. Global Authoring Practices for the Mobile Web, 2010.

[Paw04] Sourabh A. Pawar. A Common Software Development Framework For Coordinating Usability Engineering and Software Engineering Activities. jun 2004.

[PB13] Dennis Pagano and Bernd Brügge. User involvement in software evolution practice: a case study. pages 953–962, may 2013.

[PN08] Volker Paelke and Karsten Nebe. Integrating agile methods for mixed reality design space exploration. In *Proceedings of the 7th ACM conference on Designing interactive systems - DIS '08*, pages 240–249, New York, New York, USA, feb 2008. ACM Press.

[Pow17] Powtoon. PowToon - Brings Awesomeness to Your Presentations, 2017.

[Pra17] K. Siva Prasad Reddy. *Beginning Spring Boot 2: Applications and microservices with the Spring framework*. 2017.

[Pre13] Bernhard Preim. *Entwicklung interaktiver Systeme: Grundlagen, Fallbeispiele und innovative Anwendungsfelder*, volume 9. Springer-Verlag, 2013.

[PRS+94] Jenny Preece, Yvonne Rogers, Helen Sharp, David Benyon, Simon Holland, and Tom Carey. Human-Computer Interaction. apr 1994.

[Que01] Whitney Quesenbery. Building a Better Style Guide. *Style (DeKalb, IL)*, 2001.

[Que03] Whitney Quesenbery. Dimensions of Usability: Defining the Conversation, Driving the Process. *Proceedings of the Usability Professional's Association (UPA) conference on Ubiquitous Usability*, 2003.

[RF07] Michael Richter and Markus Flückiger. *Usability Engineering kompakt: benutzbare Software gezielt entwickeln*. 2007.

[RF13] Michael Richter and Markus D. Flückiger. *Usability Engineering kompakt: Benutzbare Produkte gezielt entwickeln*. 2013.

[RH09] Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009.

[RH13] Amelie Roenspieß and Michael Herczeg. Interaktive und rollenspezifische Styleguides im Usability-Engineering. *Mensch & Computer*, pages 285–288, 2013.

[RM08] Jo Rabin and Charles McCathieNevile. Mobile Web Best Practices 1.0, 2008.

[Ros99] Integrating Development of Task and Object Models. *Communications of the ACM*, 42(1):49–56, 1999.

[Sch04] Eric Schaffer. *Institutionalization of Usability: A Step-by-step Guide*. Addison-Wesley Professional, 2004.

[SDKP06] Ahmed Seffah, Mohammad Donyaee, Rex B. Kline, and Harkirat K. Padda. Usability measurement and metrics: A consolidated model. *Software Quality Journal*, 14(2):159–178, 2006.

[SFM05] Kenia Sousa, Elizabeth Furtado, and Hildeberto Mendonça. UPi. In *Proceedings of the 2005 Latin American conference on Human-computer interaction - CLIHC '05*, pages 76–87, New York, New York, USA, oct 2005. ACM Press.

[SH06] Florian Sarodnick and Brau Henning. Amazon.de: Methoden der Usability Evaluation: Wissenschaftliche Grundlagen und - Florian Sarodnick, Henning Brau: Bücher, 2006.

[SH10] Martin Schrepp and Theo Held. Anwendung von GOMS-Analysen und CogTool in der Design-Praxis. In *Mensch & Computer 2010: Interaktive Kulturen, Interdisziplinäre Fachtagung, 12.-15. September 2010, Duisburg, Germany*, jan 2010.

[Sha85] Brian Shackel. Human Factors and Usability - Whence and Whither. pages 13–31, sep 1985.

[Sha86] B Shackel. Ergonomics in design for usability. pages 44–64, dec 1986.

[Sha09] Brian Shackel. Usability - Context, framework, definition, design and evaluation. *Interacting with Computers*, 21(5-6):339–346, 2009.

[SHBA+14] Mirjam Seckler, Silvia Heinz, Javier A. Bargas-Avila, Klaus Opwis, and Alexandre N. Tuch. Designing usable web forms. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, pages 1275–1284, New York, New York, USA, 2014. ACM Press.

[SHH+15] Maria Shitkova, Justus Holler, Tobias Heide, Nico Clever, and Jörg Becker. Towards Usability Guidelines for Mobile Websites and Applications, 2015.

[Shn97] Ben Shneiderman. Designing the User Interface: Strategies for Effective Human-Computer Interaction. jul 1997.

[SPCJ13] Ben Shneiderman, Catherine Plaisant, Maxine Cohen, and Steven Jacobs. *Designing the User Interface: Pearson New International Edition: Strategies for Effective Human-Computer Interaction*. Pearson Education Limited, 2013.

[SRP07] Helen Sharp, Yvonne Rogers, and Jenny Preece. *Interaction design: beyond human-computer interaction*, volume 11. 2007.

[Sun99] Inc Sun Microsystems. Java Look and Feel Design Guidelines. *Design*, 1999.

[Tah16] Hamed Taherdoost. Validity and Reliability of the Research Instrument; How to Test the Validation of a Questionnaire/Survey in a Research. *SSRN Electronic Journal*, 2016.

[Tei12] Pedro Teixeira. *Professional Node.js*. 2012.

[Tho98] Rita Leigh Thomas. Elements of performance and satisfaction as indicators of the usability of digital spatial interfaces for information-seeking: implications for isla. jan 1998.

[Tid10] Jenifer Tidwell. *Designing Interfaces*. 2010.

[TM11] Antero Taivalsaari and Tommi Mikkonen. The web as an application platform: The Saga continues. In *Proceedings - 37th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2011*, pages 170–174, 2011.

[Tod] Todd Motto. Ultimate guide to learning AngularJS in one day.

[Tra97] Noam Tractinsky. Aesthetics and apparent usability. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '97*, pages 115–122, New York, New York, USA, mar 1997. ACM Press.

[UXS] UXSolution::A datepicker for twitter bootstrap.

[Van01] Mark Van Harmelen. Object modeling and user interface design: designing interactive systems. jan 2001.

[VB08] Viswanath Venkatesh and Hillol Bala. Technology acceptance model 3 and a research agenda on interventions. *Decision Sciences*, 2008.

[VDC00] V. Venkatesh, Fred D Davis, and Sam M Walton College. Theoretical Acceptance Extension Model : Field Four Studies of the Technology Longitudinal. *Management science*, 2000.

[vdHdJS03] Maaike J. van den Haak, Menno D.T. de Jong, and Peter Jan Schellens. Retrospective vs. concurrent think-aloud protocols: testing the usability of an online library catalogue, sep 2003.

[VM] Jean Vanderdonckt and Francisco Montero Simarro. Generative Pattern-Based Design of User Interfaces.

[Vog01] Thomas Vogt. Difficulties in Using Style Guides for Designing User Interfaces. In *Tools for Working with Guidelines*, pages 197–208. Springer London, London, 2001.

[VTS+16] Viswanath Venkatesh, James Y L Thong, Business Statistics, Xin Xu, and Technology Acceptance. Unified Theory of Acceptance and Use of Technology: A Synthesis and the Road Ahead. *Jais*, 2016.

[WBH88] John Whiteside, John Bennett, and Karen Holtzblatt. Usability Engineering: Our Experience and Evolution. In *The Handbook of Human computer Interaction*, pages 791–817. 1988.

[WCFD+12] Roel Wieringa, Nelly Condori-Fernandez, Maya Daneva, Bela Mutschler, and Oscar Pastor. Lessons learned from evaluating a checklist for reporting experimental and observational research. In *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM '12*, page 157, New York, New York, USA, sep 2012. ACM Press.

[WF87] Terry Winograd and Fernando Flores. Understanding Computers and Cognition: A New Foundation for Design. sep 1987.

[Wie10] Roel Wieringa. Design science methodology. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - ICSE '10*, volume 2, page 493, New York, New York, USA, may 2010. ACM Press.

[Wil04] Chauncey Wilson. Guidance on Style Guides: Lessons Learned, 2004.

[Wol] Natalie Woletz. Evaluation eines User-Centred Design-Prozessassessments : empirische Untersuchung der Qualität und Gebrauchstauglichkeit im praktischen Einsatz.

[YM18] Yoon and Mrowetz. Stylify me, 2018.

Appendix

Pre-observation questionnaire

1. Alter
   - unter 20
   - 20 - 30 Jahre
   - 31 - 40 Jahre
   - 41 - 60 Jahre
   - über 60
2. Programmiererfahrung in Jahren
3. Was ist deine Rolle im Projekt MineralBay? Wie lange bist du schon dabei?
4. Wie schätzt du dein generelles Wissen in Software Engineering auf einer Skala von 1-7 ein?
5. Wie schätzt du dein generelles Wissen bezüglich Usability Design auf einer Skala von 1-7 ein?
6. Wie schätzt du dein generelles Wissen bezüglich Styleguides / Pattern Libraries auf einer Skala von 1-7 ein?
7. Wie oft verwendest du Styleguides / Pattern Libraries?
   - gar nicht
   - selten (1x im Jahr)
   - manchmal (mind. 1x im Jahr)
   - häufig (mind. 1x alle zwei Monate)
   - regelmäßig (mind. 1x im Monat)
8. In welchem Zusammenhang hast du ggf. Styleguides verwendet?
9. Welche Inhalte erwartest du dir vom Styleguide?

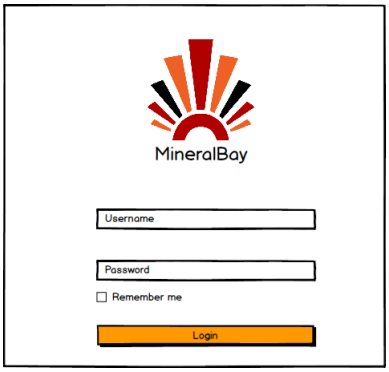
10. Wie müsste ein solches Styleguide gestaltet sein, damit du es in deinem Arbeitsalltag nutzt?

11. Siehst du einen grundsätzlichen Bedarf nach einem Styleguide im MineralBay?

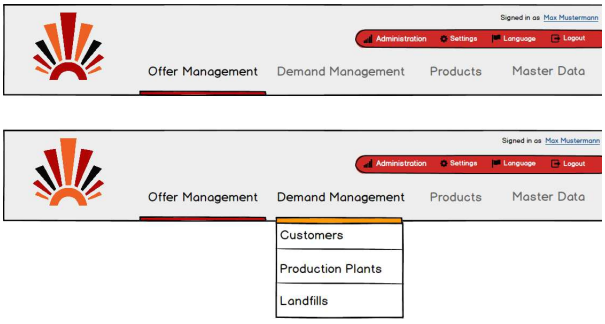
Tasks of the contextual inquiry

1. Datepicker
   Unify one date picker in the project using the date picker from the style guide.

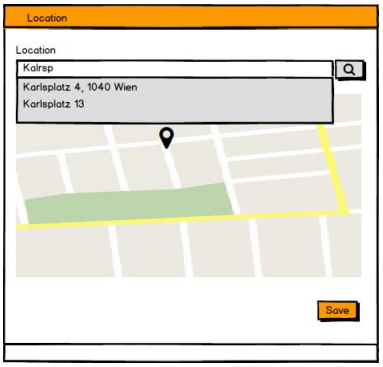
2. Floating Labels
   Design the login page according the following mockup. Use floating labels for the input fields.



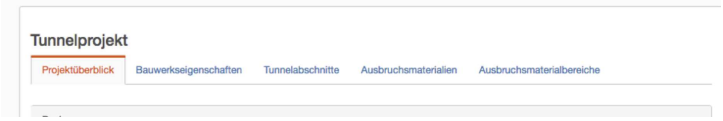
3. Replace the side navigation with a top navigation bar
   Design the top navigation bar according the following mockups. The MineralBay logo is clickable and links to the homepage. It changes the colors on hover.



4. Loading Locations
   Create a typeahead for the address input field and retrieve matches using the $http service. Design the address section according the the following mockup.

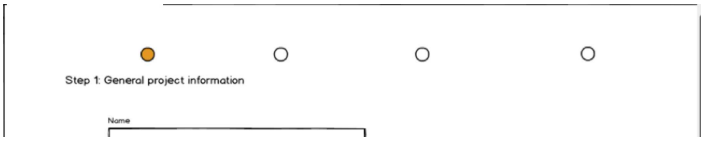


5. Tabs
   Adjust the styling of all tabs in the project according to the mockup.

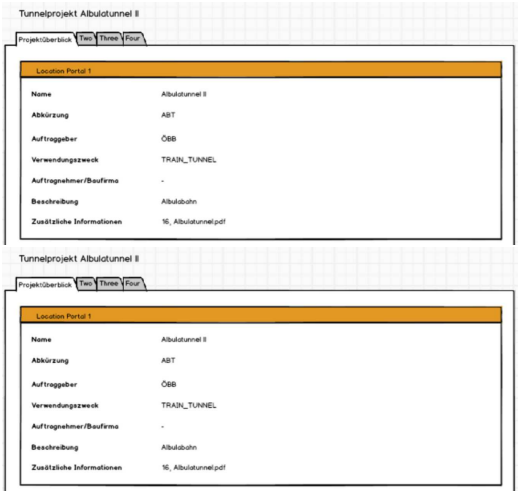


6. Inline Validation
   Check the validity of the user's inputs as the user progresses through the form. Only indicate warning and error states. To show error messages make use of „Tooltips".

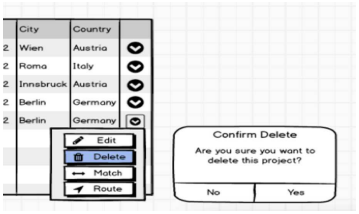
7. Wizards
   The input of a tunnel project should be structured with a wizard. Divide the input data into logical sections and use a wizard to navigate between them. For editing and displaying a tunnel project, the view with the tabs should be preserved.



8. Section Editing
   Make the panels of a tunnel project editable. Get rid of the save and back buttons at the end of the mask and save the information after editing a panel. Show the button for editing a panel at mouseover. Make use of the „Xeditable" component.



9. Dropdown buttons
   Use dropdown buttons to display action buttons inside the table of the tunnel projects.



Post-observation questionnaire

1. Was hat dir an den Aufgaben gefallen? - nicht gefallen? Wie schwer fandest du die Aufgaben? Waren sie verständlich genug?
2. Wie fandest du die Hauptnavigation? Wie fandest du die Kategorisierung der Hauptnavigation?
3. Fandest du die Seitennavigation hilfreich? Was hat dir gefallen bzw. nicht so gut gefallen?
4. Wie fandest du die Patterndetailansicht? – die Darstellung des Codes?
5. Wie fandest du die Performance der Seite?
6. Findest du, dass der Styleguide deine usability-technischen Fragen beantworten kann?
7. Gibt es fehlende/obsolete Widgets? Wenn ja, welche?
8. Denkst du, dass dich der Styleguide bei deiner Entwicklungsarbeit unterstützen würde oder würde er eher deine Entwicklungsroutine stören?
9. Was hat dir am Styleguide gefallen? Was hat dir nicht gefallen?

Results of the contextual inquiries

Developer 1

Results of the pre-observation questionnaire

1. unter 20
2. 10 Jahre an Programmiererfahrung
3. Rollen: Entwickler, Requirement Engineer seit 2 Jahren
4. Gute (5) Skills in Software Engineering
5. Wenig (2-3) Wissen in Usability Design
6. Wenig (2) Wissen bzgl. Styleguides / Pattern Libraries
7. selten
8. Hat Bootstrap bei einigen universitären Projekten verwendet.

9. Erwartet sich, dass die zu verwendenden Komponenten beschrieben sind.
10. Einfache und übersichtliche Struktur; einfach sich zurecht zu finden.
11. ja

Sequence of tasks
1-2-7-4-5-6-3-8-9 (siehe 8)

Additionally tools used

- Websuche: Dokumentation des Bootstrap-Datepickers
- Websuche: Zentrieren von Images in CSS

Results of the post-observation questionnaire

1. Die erste Aufgabe hätte recht einfach gehen sollen. Es war ein Problem mit den Datentypen. Der weitere Verlauf hatte nicht wirklich was mit dem Styleguide zu tun. Die zweite Aufgabe war recht gut. Die Dritte war sehr kompliziert bzw. aufwändig zu lösen, aber ich denke, dass es hier die Absicht war auch mal eine komplizierte Aufgabe zu lösen.
2. Hauptnavigation: Verständlich, wenn man mal mit Bootstrap gearbeitet hat. Bei der Introduction wäre eine kurze Erklärung der Hauptkategorien nicht schlecht.
3. Seitennavigation: sehr angenehm, wenn man mal rausgefunden hat, was sie macht. Das Einzige ist, dass einige Kategorien recht lang sind (schwer zum Scrollen). Hier wären Unterkategorien sinnvoll bzw. würden auch Brotkrümmel gehen.
4. Es ist recht übersichtlich. Ich fand es auch angenehm, dass man eine Preview hat.
5. Performance: wäre mir nichts aufgefallen, das zu langsam wäre.
6. Kommt sehr stark auf die Aufgabe an, nicht nur auf den Styleguide. Bei speziellen Sachen, wie man z.B: ein Bild zentriert oder so eher nicht. Aber sonst ist er eine extreme Hilfe.
7. Nein
8. Unterstützen: Vor allem bei Sachen, die ich neu mache.
9. Der Styleguide macht sehr vieles einheitlich, z.b hatte jeder Entwickler einen eigenen DatePicker verwendet. Bei Komponenten, die an mehreren Stellen / bei mehreren Kategorien vorkommen, könnte man Verweise machen. Bei allen Widgets sollte eine Information zur Installation dabei sein. Muss jetzt keine Anleitung sein, kann auch eine Bemerkung sein, dass man auf der verlinkten Seite nachschauen soll.

Es war für mich nicht verständlich, wie weit die CSS-Sachen schon im Projekt inkludiert sind und was ich noch einbauen muss. Ich findes es sehr hilfreich, dass auch verschiedene Arten gezeigt werden, wie man Forms programmiert. Vor allem für jemanden, der nicht so viel UI bzw. AngularJS gemacht hat. UX-Guidelines: Hier habe ich kurz reingeschaut, wäre vor allem bei komplizierteren Eingabemasken hilfreich, z.B. gibt es hier Hinweise wie man Labels machen soll, wie man die Seite aufbauen soll, etc. Das kann bei komplizierteren Eingabemasken sehr gut helfen. Die praktischen Beispiele findet ich auch gut.

Developer 2

Results of the pre-observation questionnaire

1. 20-30 Jahre
2. 9 Jahre an Programmiererfahrung
3. Rollen: Entwickler seit einem Jahr
4. Gute (5) Skills in Software Engineering
5. Wenig (2) Wissen in Usability Design
6. Wenig (2) Wissen bzgl. Styleguides / Pattern Libraries
7. regelmäßig
8. bei der Arbeit
9. einheitliche Gestaltungsrichtlinien
10. Verständlich, mit Beispielen und Code Snippets
11. ja

Sequence of tasks

2-6-3-1-4-6-5-7-8-9 (siehe 8)

Additionally tools used

- Websuche: Floating Labels
- Websuche: Überschreiben von Less-Variablen in Bootstrap.

Occurred problems

- Das Projekt musste neu aufgesetzt werden

- MineralBay konnte am Anfang nicht gestartet werden.

Results of the post-observation questionnaire

1. Aufgaben: 1-2 waren verständlich, zum Rest bin ich nicht gekommen.
2. Hauptnavigation: sehr gut und einfach. Ist mir familiär vorgekommen, weil ich auch Bootstrap verwende. Bei Introduction würde ich mir noch eine Beschreibung erwarten, was das alles ist, bzw. ein CSS zum Herunterladen. Auch eine Installationsanleitung für Angular, weil es für Angular gemacht ist.
3. Seitennavigation: gut
4. gut und übersichtlich
5. ausreichend gut
6. Der Styleguide konnte mir nicht beantworten, wann man eine Warning bzw. ein Error verwenden soll. Ansonsten konnte mir der Styleguide alle Fragen beantworten.
7. für mich auf jeden Fall
8. Definitiv unterstützen, weil man Sachen kopieren kann. Das geht schnell.
9. Meiner Meinung nach fehlt ein CSS-Downloadable bei Introduction. Ansonsten fand ich den Styleguide recht hilfreich.

Developer 3

Results of the pre-observation questionnaire

1. 20-30 Jahre
2. 7 Jahre an Programmiererfahrung
3. Rollen: Entwickler seit 2 Jahren
4. Gute (5) Skills in Software Engineering
5. Moderates (3-4) Wissen in Usability Design
6. Moderates (4) Wissen bzgl. Styleguides / Pattern Libraries
7. regelmäßig
8. bei der Web-Entwicklung
9. Dass, die Komponenten definiert sind und auch Farben für die Buttons, ...

10. Wie im Bootstrap, mit einer Komponentenübersicht, Beschreibung and Farbfestlegungen (info, warning,..) für Meldungen, Buttons,...

11. ja

Sequence of tasks
1-2-3-9-8-7-6-5-4 (siehe 8)

Additionally tools used

- Websuche: Lokation der CSS-Datei im Angular
- Websuche: Zentrieren von Images in CSS
- Websuche: Ändern von Farben in Bootstrap

Occurred problems
MineralBay konnte am Anfang nicht gestartet werden.

Results of the post-observation questionnaire

1. Waren ok und verständlich.

2. Hauptnavigation: gut; vor allem auch, dass die Angularsachen und die Bootstrapkomponenten getrennt sind.

3. Seitennavigation: habe ich nicht wirklich verwendet, weil ich lieber gleich gesucht habe. Allerdings finde ich die schon nützlich.

4. Unterscheidung der Codearten: DatePicker, hier gab es einen Bug, dass wenn man vom Markup auf Preview wechselt, der falsche Code angezeigt wird. Das hat mich verwirrt. Ich würde es besser finden, wenn man Markup und Preview nicht trennen würde.

5. War gut.

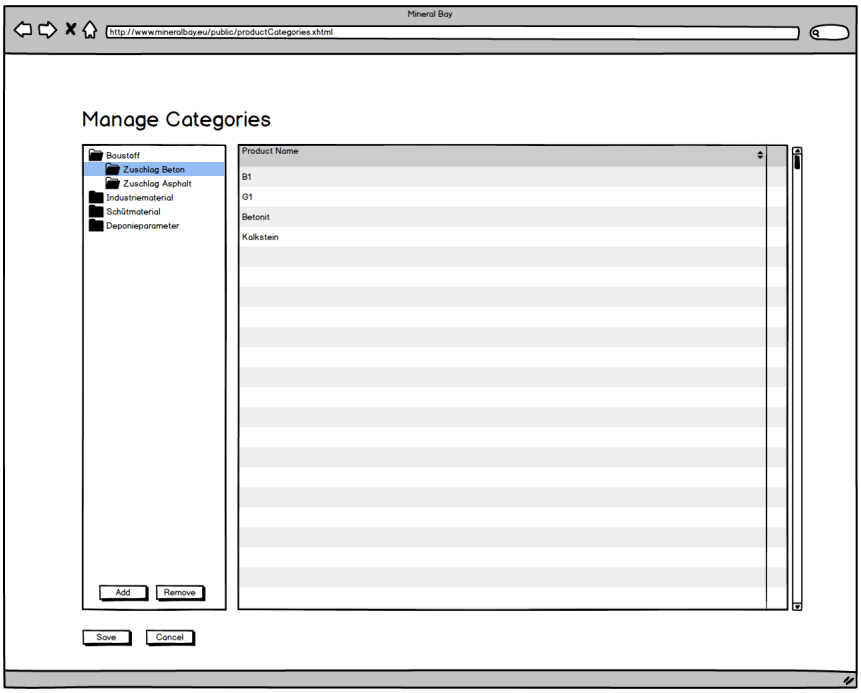
6. Es sind keine usability-technischen Fragen aufgetaucht. Habe die Guidelines bemerkt und finde, dass ich hier schon nachschauen würde, wenn ich eine neue Form anlege. Es ist sehr viel Information zum Einlesen. Man könnte sich das mal durchlesen, wenn man neu im Projekt ist.

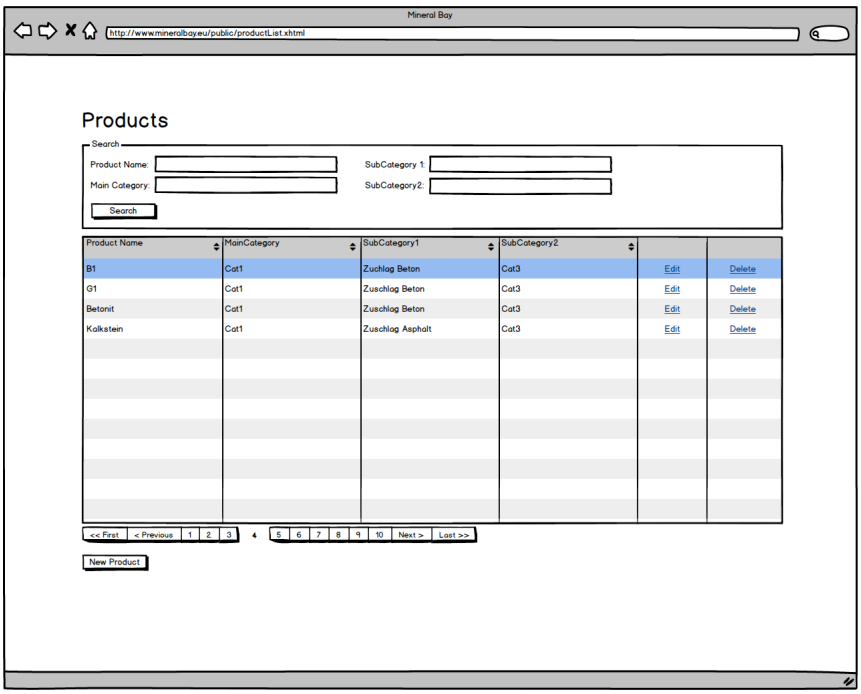
7. Schauen sehr vollständig aus.

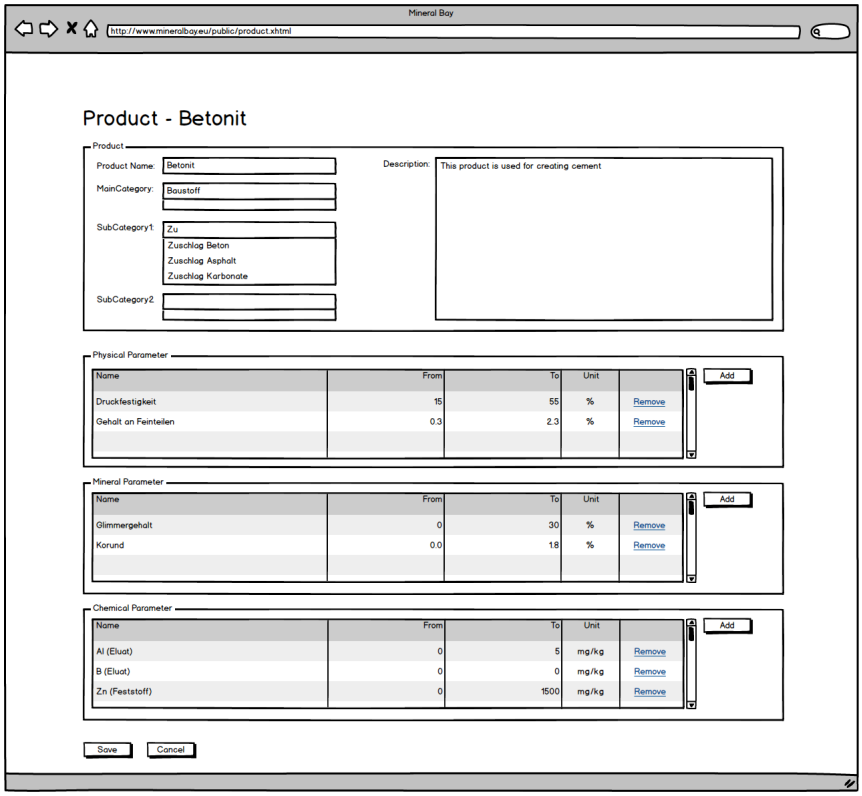
8. Denke schon, es wäre eine zentrale Anlaufstelle, z.B. wenn ich eine Pagination einbauen möchte.

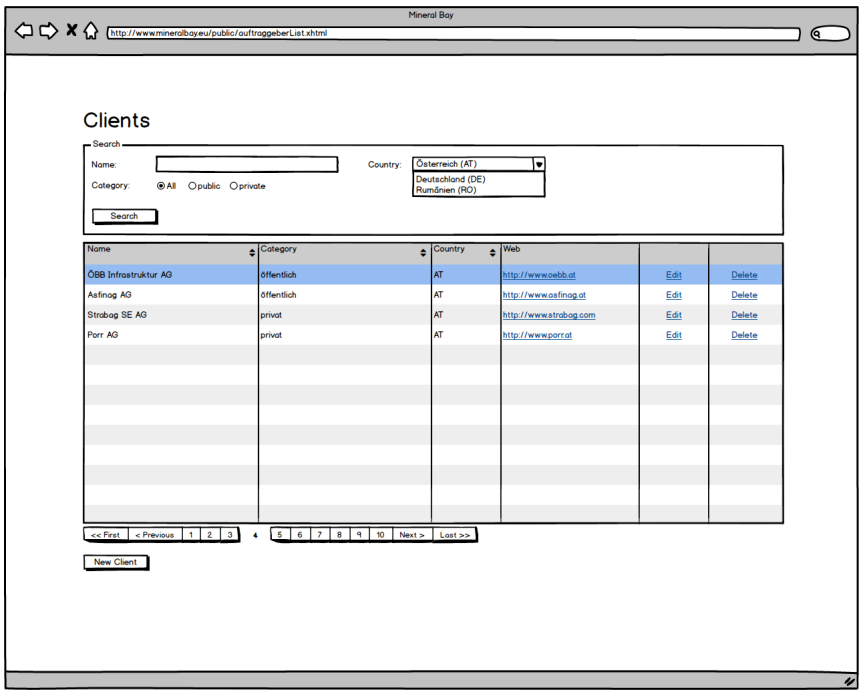
9. Eine Anleitung wie man CSS-Code ändert hat gefehlt. Was z.B. die einzelnen CSS-Dateien sind, die man im Projekt findet. Wo im Projekt die Css-Datei ist, in die der CSS-Code einzufügen ist. Ich würde den Unterschied zwischen Less und CSS im Bootstrap erklären und welche Variante wir implementiert haben. Installationsanleitungen bei allen Widgets hinzufügen bzw. vereinheitlichen. Verlinkung zwischen den Komponenten und den Informationen/Guidelines fehlt.

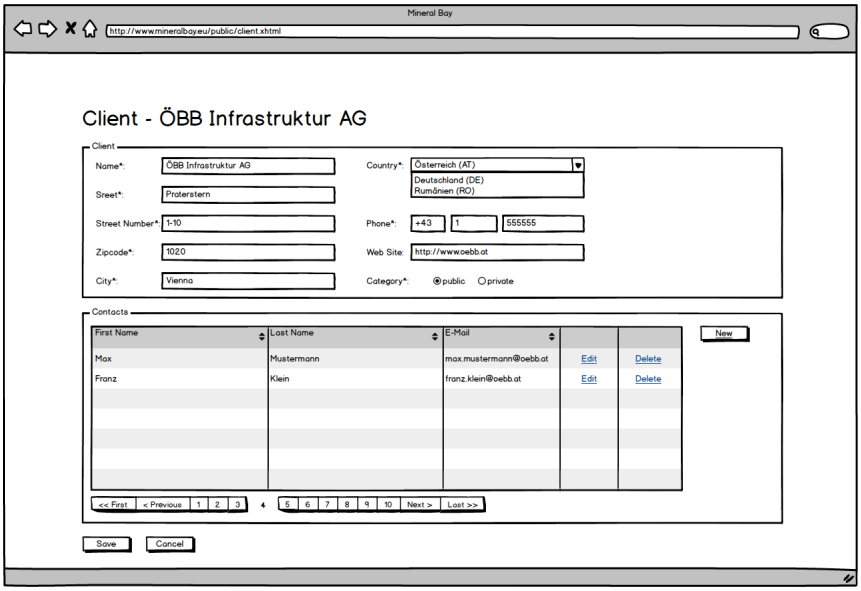
Initial Wireframes of MineralBay

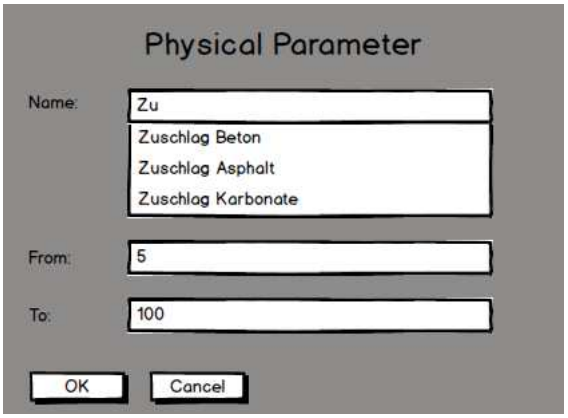


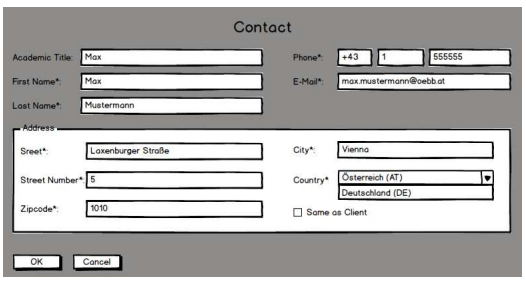












Questionnaire for the developers of MineralBay

Fragebogen im Rahmen der Anforderungsanalyse zur Erstellung eines Styleguides für das Projekt MineralBay

Dieser Fragebogen richtet sich an die Entwickler der Anwendung MineralBay

* Required

1. **Geschlecht** *
   Mark only one oval.
   - ◯ männlich
   - ◯ weiblich

2. **Alter** *
   Mark only one oval.
   - ◯ unter 20
   - ◯ 20-30 Jahre
   - ◯ 31-40 Jahre
   - ◯ 41-60 Jahre
   - ◯ über 60

3. **Programmiererfahrung in Jahren** *

4. **Wie schätzt du dein generelles Wissen in Software Engineering auf einer Skala von 1-7 ein?** *
   Je höher die angegebene Zahl, desto höher schätzt du dein Fachwissen ein.
   Mark only one oval.

1	2	3	4	5	6	7
◯	◯	◯	◯	◯	◯	◯

5. **Wie schätzt du dein generelles Wissen bezüglich Usability Design auf einer Skala von 1-7 ein?** *
   Je höher die angegebene Zahl, desto höher schätzt du dein Fachwissen ein.
   Mark only one oval.

1	2	3	4	5	6	7
◯	◯	◯	◯	◯	◯	◯

6. **Wie schätzt du dein generelles Wissen bezüglich Styleguides / Pattern Libraries auf einer Skala von 1-7 ein?** *
   Je höher die angegebene Zahl, desto höher schätzt du dein Fachwissen ein.
   Mark only one oval.

1	2	3	4	5	6	7
◯	◯	◯	◯	◯	◯	◯

7. **Wie oft verwendest du Styleguides / Pattern Libraries?** *
   Mark only one oval.
   - ◯ Gar nicht
   - ◯ Selten (1x im Jahr)
   - ◯ Manchmal (mind. 1x im Jahr)
   - ◯ Häufig (mind. 1x alle zwei Monate)
   - ◯ Regelmäßig (mind. 1x im Monat)

8. **In welchem Zusammenhang hast du ggf. Styleguides verwendet?**

9. **Welche Inhalte erwartest du dir vom Styleguide?** *

10. **Wie müsste ein solches Styleguide gestaltet sein, damit du es in deinem Arbeitsalltag nutzt?** *

11. **Welche der folgenden Aussagen trifft deine Meinung über Styleguides?** *

Mark only one oval per row.

	trifft völlig zu	trifft zu	teils/teils	trifft nicht zu	trifft gar nicht zu
Ein Styleguide würde vieles komfortabler machen.	◯	◯	◯	◯	◯
Ein Styleguide würde mich bei meiner Entwicklungsarbeit unterstützen.	◯	◯	◯	◯	◯
Ich denke, dass die Nutzung eines Styleguides immer mit einem gewissen Risiko verbunden ist.	◯	◯	◯	◯	◯
Ich denke, dass ein Styleguide Gefahren für mich birgt.	◯	◯	◯	◯	◯
Ein Styleguide würde meine Entwicklungsroutine stören.	◯	◯	◯	◯	◯
Ein Styleguide würde mir mehr Nachteile als Vorteile bringen.	◯	◯	◯	◯	◯

12. **Siehst du einen grundsätzlichen Bedarf nach einem Styleguide?** *

Mark only one oval.

- ◯ Ja
- ◯ Nein
- ◯ Other: _____

13. **sonstige Anmerkungen**

Questionnaire for the potential users of MineralBay

Fragebogen im Rahmen der Anforderungsanalyse zur Erstellung eines Styleguides für das Projekt MineralBay

* Required

Vorstellungsvideo

http://youtube.com/watch?v=MDhqOfSYNnQ

1. **Geschlecht** *
   Mark only one oval.
   - ◯ männlich
   - ◯ weiblich

2. **Alter** *
   Mark only one oval.
   - ◯ unter 20
   - ◯ 20-30 Jahre
   - ◯ 31-40 Jahre
   - ◯ 41-60 Jahre
   - ◯ über 60

3. **Für welche Art von Unternehmen sind Sie zur Zeit tätig?** *
   Mark only one oval.
   - ◯ Öffentlicher Auftraggeber
   - ◯ Baufirma
   - ◯ Ingenieurbüro
   - ◯ Rohstoffgewinnungs- bzw. -verarbeitungsbetrieb
   - ◯ Deponiebetreiber
   - ◯ Selbstständig
   - ◯ Student
   - ◯ Other: _____

4. **Was ist ihr derzeitiger Aufgabenbereich im Unternehmen?**
Mark only one oval.

- ◯ Geschäftsführung
- ◯ Informationstechnologie (IT)
- ◯ Forschung & Entwicklung (F & E)
- ◯ Produktion
- ◯ Marketing
- ◯ Vertrieb
- ◯ Service
- ◯ Finanzen und Controlling
- ◯ Human Resources (HR)
- ◯ Logistik
- ◯ Other: _____

5. **Was ist Ihre höchste abgeschlossene Ausbildung?** *
Mark only one oval.

- ◯ Pflichtschule
- ◯ Fachschule oder Lehre
- ◯ Matura
- ◯ Hochschulabschluss (Universität oder Fachhochschule)

6. **Seit wann in etwa nutzen Sie das Internet?** *

Mark only one oval.

- 1985
- 1986
- 1987
- 1988
- 1989
- 1990
- 1991
- 1992
- 1993
- 1994
- 1995
- 1996
- 1997
- 1998
- 1999
- 2000
- 2001
- 2002
- 2003
- 2004
- 2005
- 2006
- 2007
- 2008
- 2009
- 2010
- 2011
- 2012
- 2013
- 2014
- 2015
- 2016
- 2017

7. **Welche Browser verwenden Sie generell?** *

Check all that apply.

- [ ] Google Chrome
- [ ] Internet Explorer
- [ ] Safari
- [ ] Mozilla Firefox
- [ ] Opera
- [ ] Microsoft Edge
- [ ] Other: _____

8. **Wie viele Stunden verbringen Sie die Woche im Internet?** *
Mark only one oval.

- ◯ 0 - 5 Stunden
- ◯ 6 - 15 Stunden
- ◯ 16 - 25 Stunden
- ◯ 26 - 35 Stunden
- ◯ 36 - 45 Stunden
- ◯ 46 - 55 Stunden
- ◯ 56 - 70 Stunden
- ◯ über 70 Stunden

9. **Wann verbringen Sie typischerweise Zeit im Internet?** *
Check all that apply.

- ☐ bei der Arbeit
- ☐ zu Hause
- ☐ unterwegs
- ☐ Other: _____

10. **Welche Geräte benutzen Sie um ins Internet zu gelangen?** *
Check all that apply.

- ☐ PC oder Laptop
- ☐ Smartphone
- ☐ Tablet
- ☐ Other: _____

11. **Welche Geräte benutzen Sie zur Verrichtung Ihrer Arbeit?** *
Check all that apply.

- ☐ PC oder Laptop
- ☐ Smartphone
- ☐ Tablet
- ☐ Other: _____

12. **Auf welchem Gerät/welchen Geräten würden Sie MineralBay nutzen?** *
Check all that apply.

- ☐ PC oder Laptop
- ☐ Smartphone
- ☐ Tablet
- ☐ Other: _____

13. **Haben Sie Lieblingsseiten? Wenn ja, welche?** *
Eingabe in der Form: google.com, orf.at,..

_____

14. **Welche der folgenden Aussagen trifft auf Sie zu?** *

Mark only one oval per row.

	trifft gar nicht zu	trifft nicht zu	teils/teils	trifft zu	trifft völlig zu
Ich mache mir oft Sorgen darüber, dass mich neue technische Geräte überfordern könnten.					
Wenn ich ein neues technisches Gerät/eine neue Software verwenden soll, bin ich erst mal misstrauisch.					
Die Vorstellung, bei der Verwendung technischer Anwendungen etwas falsch zu machen, macht mir Angst.					
Im Laufe meines Lebens habe ich mir viel technisches Wissen angeeignet					
Wenn ein neues technisches Gerät/eine neue Anwendung auf den Markt kommt, informiere ich mich darüber.					
Ich versuche immer aktuelle Informationen über neue technische Entwicklungen zu bekommen.					

15. **Wie schätzen Sie Ihr generelles Wissen bezüglich des Internets auf einer Skala von 1-7 ein?**

Je höher die angegebene Zahl, desto höher schätzen Sie Ihr Fachwissen ein.

Mark only one oval.

	1	2	3	4	5	6	7	
Basiswissen: z.B.: Google-Suche								Experte: z.B.: Verwenden von Webapplikationen, Webservices,..

16. **Warum würden Sie MineralBay verwenden und welche der folgenden Aussagen treffen ihrer Meinung nach zu?** *

Mark only one oval per row.

	trifft gar nicht zu	trifft nicht zu	teils/teils	trifft zu	trifft völlig zu
Diese Anwendung würde vieles komfortabler machen.					
Ich denke, dass die Nutzung dieser Anwendung immer mit einem gewissen Risiko verbunden ist.					
Diese Anwendung würde meine Alltagsroutine stören.					
Diese Anwendung würde mir mehr Nachteile als Vorteile bringen.					

17. **Welche drei Funktionen würden Sie sich im MineralBay wünschen?**

18. **Sehen Sie einen grundsätzlichen Bedarf nach einer solchen Anwendung (bei Ihnen selbst, bei Kollegen, in Ihrem Unternehmen)?** *

19. **Name und Kontaktdaten**

Ihre Daten werden selbstverständlich anonymisiert ausgewertet. Wir würden uns dennoch freuen, wenn Sie hier Ihre Kontaktdaten für eventuelle Rückfragen zur Verfügung stellen würden.

20. **Sonstige Anmerkungen**

Powered by Google Forms