Phil Diegmann

**Bachelorarbeit**
**im Fach Allgemeine Wirtschaftsinformatik**

# Systematic Development of mHealth Apps: Lessons Learned During Development of a Mobile Frontend for ePill

Themensteller: Jun.-Prof. Dr. Ali Sunyaev

Vorgelegt in der Bachelorprüfung
im Studiengang Wirtschaftsinformatik
der Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Universität zu Köln

Köln, August 2013

**Table of Contents**

## Index of Abbreviations

| | |
|---|---|
| app | Application |
| app user | intended audience for the app |
| CDN | Content Delivery Network. Supports high availability and performance for static content on the internet |
| CSS | Cascading Style Sheets. A language used to style web pages |
| DNS | Domain Name System. Used to translate domain names into IP-Addresses |
| eHealth | "a paradigm involving the concepts of health, technology, and commerce, with commerce and technology as tools in the service of health"[1], belonging to the field of telehealth.[2] |
| ePill | a patient-centered health IT service which offers information on pharmaceuticals and aggregation of data in context |
| framework | can contain source code, tools and libraries, which together provide specific or common but abstracted functionality |
| frontend | visible user interface for the app user |
| HECAT | Health Education Curriculum Analysis Tool[3] |
| HIT | abbreviation for Health Information Technology |
| HTML | HyperText Markup Language, a markup language to design web pages |
| IDE | abbreviation for Integrated Development Environment |
| JSON | JavaScript Object Notation, represents data structures human-readable |
| mHealth | "medical and public health practice supported by mobile devices, such as mobile phones, patient monitoring devices, personal digital assistants (PDAs), and other wireless devices"[4], also known as m-Health |
| mHealth apps | "aim at providing seamless, global access to tailored health IT services and have the potential to alleviate global health burdens"[5] |

---

1 Martínez-Pérez, de la Torre-Díez, Isabel, López-Coronado (2013), p. 2

2 cf. Martínez-Pérez, de la Torre-Díez, Isabel, López-Coronado (2013), p. 2

3 http://www.cdc.gov/HealthyYouth/HECAT/

4 World Health Organization (2011) cited by Martínez-Pérez, de la Torre-Díez, Isabel, López-Coronado (2013), p. 2

5 Dehling, Sunyaev (2013), p. 1

| | |
|---|---|
| MVC | Model-View-Controller. A software architecture pattern which separates logic and user interfaces. Models are representatives of data structures. Views contains the user interface definitions and controllers contains the application logic |
| information security | Prevention from unauthorized access to information. |
| NDK | Native Development Kit. Bundled software and tools which enables the developer to implement programs on native-code languages[6] |
| OS | Operating System |
| SDK | Software Development Kit. Bundled software and tools for developing with or for a specified OS or framework |
| sensitive information | information, which is personal. Can be related to financial-, health- or otherwise personal relevant information[7] |
| telehealth | delivery of medical- or health-related information or services via telecommunication technologies |
| usability | "extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use"[8] |
| use value | the utility of consuming a good or service |
| user interface | for humans visible controls and layout of an application |
| W3C | World Wide Web Consortium[9] |

6 cf. http://developer.android.com/tools/sdk/ndk/index.html

7 Suggested by Future of Privacy Forum, Center for Democracy & Technology (2011), p. 6, although the definition varies

8 Yeh, Fontenelle (2012), p. 64 as quoted from ISO 9241-11 (1998)

9 http://www.w3.org

## Index of Tables

**Index of Illustrations**

# 1. Introduction

## 1.1 Research Problem

While it has become easy to develop a mobile health (mHealth) application (app), there is much more to it than just the aspects of the app's core functionality. Currently only very few guidelines, best practices and systematic development approaches for mobile app development can be found. Furthermore even less can be found for the specific area of mHealth apps.

Security leaks or even abuse of private and sensitive information can lead to great harm for the app user and to legal issues for the developer. Abuse of personal health related information can result in loss of reputation (e.g. sexual transmitted diseases) or financial drawbacks and decreased chances of employment (e.g. chronic diseases, genetic dispositions)[10]. With poorly developed apps, there is a chance of security leaks and hence for data abuse. Thus the risk for app users increases. A study[11] has shown that very few mHealth apps entail little or low risk for the app user. Self-publishing through modern sales channels like Google Play (http://play.google.com) or the iOS App Store (http://appstore.com) and the availability of easy-to-use Integrated Development Environments (IDEs) lower the barriers for entry. Even one-man developers or small teams are now able to publish apps easily with little development effort. Without fundamental knowledge of privacy and security aspects, there is an increase in the non-professional developmental of mobile apps with inadequate security aspects.

The usability, especially in critical situations, is another undervalued aspect in many non-professional developments. While fancy colors might look appealing to the developer himself, it might lead to confusion for the app user or even to a lack of operability for visually impaired people.[12] Also, the need for a intuitive user interface might not be considered as important as it should be.

---

10 cf. Dehling, Sunyaev (2013), pp. 6-7

11 cf. Njie (2013), pp. 19-20

12 cf. Badashian et al. (2008) p. 108

Knowledge of data privacy acts and laws is a premise for a legal, safe and fair development for the developer and the app user. Multiple layers of data privacy laws in Europe on international, national and state level require a certain legal knowledge.[13] Also, the benefit of and the need for a privacy policy seems to be ambiguous for many non-professional developers.[14]

This lack of guidelines for mobile app development and of specific guidelines for privacy and usability sensitive apps is only superficially considered by most of the literature. The beforehand highlighted aspects of usability and information security are just two of multiple possible requirements. Current research seems not to state which specific requirements, if any, distinguish mHealth apps from other apps or which are needed to be more accented.

## 1.2 Objectives of this Thesis

The purpose of this thesis is to discover, identify and report issues and challenges of the development of mHealth apps by developing a mobile frontend for the ePill system (developed by the University of Cologne, http://epill.uni-koeln.de). ePill is a patient-centered health IT service which offers information on pharmaceuticals and aggregation of pharmaceutical data in context.

During the development of a mobile frontend for ePill, all requirements can be addressed more easily than in a completely theoretical context. As a side effect, a mobile app for ePill will increase the accessibility for the ePill system in general, and thereby increase the possible user value. In critical situations in which one does not have one's desktop computer at hand, a mobile easy-to-use app can be of value.

The experiences gained during the development refer to general mobile app development,

---

[13] cf. Directive 95/46 of the European Parliament and of the Council (October, 24th 1995), Directive 2002/58 of the European Parliament and of the Council (July, 12th 2002) cited by Future of Privacy Forum, Center for Democracy & Technology (2011), p. 16

[14] cf. Njie (2013), p. 20

but also to the specific development of mHealth apps.

Mainly this thesis aims to describe the planning and the development process and discuss all discovered issues and challenges for planning and developing mHealth apps. One sub-objective is to give a short overview about the state of research on guidelines and important factors of mHealth app development. Subsequently, this thesis aims to highlight specific characteristics of mHealth apps and focus on them during the development.

# 2. The ePill System

## 2.1 The System in General

The ePill system (http://epill.uni-koeln.de) was developed by the University of Cologne to improve the readability and comprehensibility of instruction leaflets contained within the packaging of pharmaceutical drugs. Additionally ePill aims to provide further information on adverse reactions and interactions of different medical drugs. ePill emphasizes an easy readability and access to informations.
ePill is currently a prototype of a system, used only for research purposes and it is only actively used by the University of Cologne. Therefor it is only localized in German and contains only pharmaceuticals available in Germany. ePill utilizes the "GELBE LISTE PHARMINDEX"[15], provided by Medizinische Medien Informations GmbH MMI.

There are three major functions covered by the system: searching for pharmaceuticals, display information on pharmaceuticals and supplementing services.[16] The search enables the user to find corresponding pharmaceuticals depending on specified parameters in the underlying database. As an extend, the display functionality enables the user to read the leaflet information in an optimized fashion. Finally supplementing services are provided to refine the displayed information (e.g. select the level of detail of the displayed information), linking pharmaceuticals as well as other information and aggregate pharmaceutical information (e.g. interactions).

An integration and personalization depending on the current user's health records was not implemented due to the arising privacy and trust challenges.[17, 18]

The system uses a Model-View-Controller (MVC) architecture[19] and utilizes a relational

---

15 http://www.gelbe-liste.de

16 cf. for this paragraph Dehling, Sunyaev (2012a), p. 2 and Dehling, Sunyaev (2012b), p. 5

17 cf. Kaletsch, Sunyaev (2011) cited by Dehling, Sunyaev (2012a), p. 2

18 cf. Kaletsch, Sunyaev (2011), pp. 5-6

19 cf. Dehling, Sunyaev (2012a), p. 3

database as persistent data storage.[20] The data is organized in an atomic way. Products are any pharmaceuticals, which may contain specific molecules, which themselves may be related to specific adverse reactions with other molecules.
With this atomized organization of the pharmaceutical information, it becomes more easily to compare different pharmaceuticals and have very consistent information about molecules and adverse reactions for different pharmaceuticals.

## 2.2 The Web Application

The web application of the ePill system introduces itself highly customizable to the user. It offers the user the choice between a default view, a customizable view and an expert view. The default view aims to provide all necessary information in a compact way. The customizable view offers more choices for the elements to be displayed. The expert view activates all options for the most detailed information level. The pharmaceutical informations to be displayed can be fine tuned for every view. ePill offers four different presets varying from only the most basic up to all available information. These presets can be further customized by afterwards selecting or deselecting items. Additionally the font-size can be set to normal, bigger and biggest to support visually impaired users.

Three columns shape the layout. The first, leftmost column contains the main navigation for searching, pharmaceutical listings, basic functionality like help pages and settings as well as extended functionality like interactions research and adverse reaction lookup or pharmaceutical comparisons. The second, centered column contains the current content. This column has tabs, which can be assigned different contents. With this tabular layout, e.g. multiple, different search queries can easily be switched and held in parallel. The third, rightmost column can be used to dynamically display or hide specific information. Depending on the beforehand selected view, the left or right columns are hidden or visible. The website also offers the user on the pharmaceutical detail page to explain any term as well as a shortcut to the page's top.

The specific content layout is very consistent. Headlines are made salient and the ar-

---

[20] cf. Dehling, Sunyaev (2012a), p. 5 for this and the following two sentences

rangement of common sections are congruent. Changes in settings are applied with no delay and without a page reload. Any changes are made congruent with the chosen layout and other related settings.

Although this web application is not optimized for mobile applications and designed with a desktop computer in mind, it can be accessed by nearly any modern mobile computing device, like a smart phone or a tablet, and can therefor categorized as a mHealth application. This association is important to the following section, to clarify the differences between this web application and the mobile client, because with this assumption we can categorize on the same level and focus on the essential differences.

# 3. What is mHealth?

## 3.1 Definition

mHealth, also known as m-Health, is an abbreviation for mobile health and is a refinement of eHealth (or e-Health, an abbreviation for electronic health), which itself belongs to the field of telehealth.[21]

eHealth is defined as "a paradigm involving the concepts of health, technology, and commerce, with commerce and technology as tools in the service of health"[22].

Telehealth means the delivery of medical- or health-related information or services via telecommunication technologies.

mHealth in detail is defined as "medical and public health practice supported by mobile devices: such as mobile phones, patient monitoring devices, personal digital assistants (PDAs), and other wireless devices"[23]. The introduction of smart phones like the Apple iPhone and Android device led to a greater audience and the evolution of mobile tablets further increasing the audience for mHealth purposes. A study[24] relied on the Health Education Curriculum Analysis Tool (HECAT)[25] to group different mHealth apps together. This study illustrates the distribution of apps in different categories. As Tab. 3-1 illustrates most of the available apps in 2011 in the Apple App Store in the United States of America belonged to the Physical Activity area, whereas drug-related and safety-related apps (like ePill) are the least two categories in terms of app count.
From February to May of 2012, a Study by d'Heureuse et al. (2012) found several ten

---

21 cf. Martínez-Pérez, de la Torre-Díez, Isabel, López-Coronado (2013), p. 2

22 Martínez-Pérez, de la Torre-Díez, Isabel, López-Coronado (2013), p. 2

23 World Health Organization (2011) cited by Martínez-Pérez, de la Torre-Díez, Isabel, López-Coronado (2013), p. 2

24 cf. for this and the first following sentence West et al. (2012)

25 http://www.cdc.gov/HealthyYouth/HECAT/

26 Apps could be added to multiple categories

27 cf. West et al. (2012), p. 5, Table 2

| HECAT content area | n | %[26] |
|---|---|---|
| Physical Activity | 1108 | 33.21 |
| Personal health and wellness | 962 | 28.84 |
| Healthy eating | 651 | 19.51 |
| Mental and emotional health | 414 | 12.41 |
| Sexual and reproductive health | 243 | 7.28 |
| Alcohol, tobacco, and other drugs | 131 | 3.93 |
| Violence prevention and safety | 96 | 2.88 |

Tab. 3-1: HECAT Content Area App Distribution (N = 3336)[27]

thousands of apps in the Google Play Store as well as the Apple App Store just in the "Health" categories.[28] This study shows the potential of mHealth for a broader healthcare supported by mobile devices. From March to May of 2012, the total number of apps increased by an average of 6.4% (Google Play Store) and 4.5% (Apple App Store) per month.[29]

### 3.2 mHealth App Categories

Although Tab. 3-1 lists categories for mHealth apps, it focusses on content and less on the specifics for mHealth apps on other possibly important topics, such as information security or usability. Those content areas range from physical activity to health and wellness (mental, emotional, sexual or diet related) as well as for drugs and prevention. Other literature focusses on data practices and privacy risks with a more technical aspect[30].

Njie (2013) concludes that most of the mHealth apps deal in any way with directly or indirectly (e.g. via usage behavior) sensitive information. Therefor ten levels of privacy risks were developed and a sample of 43 mHealth and fitness apps were assigned to the different levels. Tab. 3-2 illustrates the characteristics of every level as well as the distribution of the 43 analyzed apps.

---

28 cf. d'Heureuse et al. (2012), p. 20, Figure 5

29 cf. d'Heureuse et al. (2012), p. 20

30 cf. for this and the following three sentences Njie (2013), pp. 13-14

The risk levels are based on the one hand on the information available to the app and on the other hand on security precautions implemented by the developer to prevent unauthorized access to this information. An important differentiation is also in the anonymity or identifiability of the information accessible by third parties. The higher the accessibility or the identifiability or the possible harm done by this information, the higher the risk level.

As stated by Istepanian, Jovanov, Zhang (2004), another categorization is possible. They categorized mHealth applications into administrative connectivity, financial connectivity or medical connectivity.[31] Because of the lack of smart phones and a far lesser availability of mobile devices in 2004 compared to today, this article cannot take the recent development in mobile devices into account, although the categorization is still appropriate. The administrative connectivity handles appointments, electronic patient records and any non-financial transactions.[32] The financial connectivity handles all financial transactions like purchases, billing or any financial services. The third connectivity, the medical connectivity, handles mobile monitoring and diagnostics.

There are three different sub-categories for mHealth applications: The content, the information security risk-level and the overall connectivity function. For the content-category as well as the connectivity-category, multiple assignments are possible. Combined these sub-categories form a specific grouping of mHealth apps. Depending on the categorization in the privacy risk, one can easily take precautionary measures. With the categorization in a HECAT content area one can identify the target audience more precisely as well as with the help of the connectivity category.

### 3.3 Classification of the ePill Web Application

ePill is to be categorized in the beforehand mentioned HECAT content areas mainly as "Alcohol, tobacco, and other drugs", because of the purpose to inform about (medical)

---

31 cf. Istepanian, Jovanov, Zhang (2004), p. 6

32 cf. for this and the two following sentences Istepanian, Jovanov, Zhang (2004), p. 13

33 cf. Njie (2013), p. 13

| Level | Risk | Characteristics | % |
|---|---|---|---|
| 9 | Highest | address, financial information, full name, sensitive or embarrassing health (or health-related) information, information that a malicious actor could use to steal or otherwise cause a user to lose money | 40 |
| 8 | High | geo-location | |
| 7 | Medium-high | DOB, ZIP code, any kind of personal medical information | |
| 6 | Medium | risk evaluated to be between level 5 and level 7 | 32 |
| 5 | Medium | email, first name, friends, interests, weight, information that is potentially embarrassing or could be used against a person (e.g., in employment) | |
| 4 | Medium | risk evaluated to be between level 5 and level 3 | |
| 3 | Medium-low | anonymized (not personally identifiable) tracking (e.g., app usage), device info, a third party knows the user is using a mobile medical app | 28 |
| 2 | Low | risk evaluated to be between level 3 and level 1 | |
| 1 | Low | any kind of anonymized data that does not include medical health-related data or personally identifiable information | |
| 0 | No | | 0 |

Tab. 3-2: Privacy Risk Levels of mHealth Apps (N = 43)[33]

drugs. Additionally, ePill informs about adverse effects and interactions, so it also belongs to the content area of "Violence prevention and safety".

The ePill web application is not connected to any electronic patient records, nor does it store any user related information such as the last searched pharmaceuticals, however it does not utilize SSL-encryption. Therefor it might not be collecting information or storing anything, but third parties could collect user specific information by monitoring.
Setting this information into context with the risk levels developed by Njie (2013), the ePill web application could be categorized as level three, if SSL-encryption would be utilized. If that would be the case, third parties could retrieve browser and OS specific information, but not data sent and retrieved with each request such as pharmaceutical information. Without encryption all data sent and retrieved is visible to possible eaves-

dropper. With information about searched pharmaceuticals, one could assemble an overall picture of the ingested medications and therefore extrapolate possible diseases. Still, all data is anonymized.
Having in mind, that ePill still is in early prototyping and assuming, that the SSL-encryption will follow, the risk is more of a medium to low level. Dealing with only anonymous data and protecting them with encryption leaves very little room for serious risks. We would therefor categorize ePill in terms of privacy risk levels as a level two.

Although ePill does not fit absolutely in any of the connectivity categories, its closest fit is within medical connectivity. Because of the aim to provide pharmaceutical (therefor medical) information, it could still be found within the medical connectivity category.

Concluding this categorization, we would suggest to categorize the ePill web application as a low privacy risk, drug and safety-related medical connectivity mHealth application. The ePill web application lacks a optimization for mobile devices but all categorizations match their definition. The HECAT content area is by definition not limited to mobile devices and privacy risks are in many ways the same for mobile apps and web applications.

### 3.4 Why is a special Focus on mHealth Apps warranted?

mHealth apps differ in some way from general (mobile) applications but also from eHealth applications. While mHealth apps can be used in many different situations and with very different intentions, the special focus on e.g. equality of all users and accessibility for all possible users are not as important for other areas of mobile apps as they are for mHealth apps.

mHealth apps are defined to "aim at providing seamless, global access to tailored health IT services and have the potential to alleviate global health burdens"[34], which means that they should be accessible by mostly all possible users. Whereas other types of apps do not necessarily need to be accessible by any user, we want to stress that accessibility does not only mean usability (especially for elderly people), but also different social layers or

[34] Dehling, Sunyaev (2013), p. 1

cultures.

Furthermore, mHealth apps deal with medical- or health-related information and have therefore to deal with sensitive information and have to address privacy risks and concerns. As pointed out by Njie (2013) and already referred to in Tab. 3-2, many mHealth apps deal with highly sensitive data and have serious privacy risks. Dehling, Sunyaev (2013) illustrates the possible damages through leaks, manipulation or loss of information.[35]

To address these concerns and issues in a mHealth project, they need to be made clear and experiences must be shared as well as interpreted. The following chapter will present all experiences made during the development of a mobile frontend for ePill in a structured way. We will list all theoretical preconditions, outline the analysis as well as the implementation of the mHealth app. Afterward, we will validate the product and give an overview about the lessons we learned.

---

[35] cf. Dehling, Sunyaev (2013), p. 7

# 4. The Development of the Mobile Client

## 4.1 Preconditions

### 4.1.1 Norms for Mobile Apps

As already mentioned, ePill is currently only used in Germany, therefore we will focus on laws applicable in Germany. These laws are namely the Telekommunikationsgesetz, the Telemediengesetz, the Directive 95/46/EG as well as the data protection act of North Rhine-Westphalia. The Telekommunikationsgesetz and Telemediengesetz are laws by state, whereas Directive 95/46/EG is an european directive, specified by the respective Member States.
German federal states have their own data protection acts. In this thesis we will focus on the data protection act of North Rhine-Westphalia as ePill is located in North Rhine-Westphalia.

As the topmost layer of laws, the Directive 95/46/EG defines more general directives. Article 4 defines national law applicable, if the natural or legal person, the controller[36], is located on a Member State's territory[37] or if any of the processing takes place on a Member State's territory[38]. Furthermore, it is required, that the controller asks the user to consent to the use and collection of data[39]. Explicitly, "data concerning health and sex life"[40] shall not be processed, only if the user consent explicitly[41], or if the processing is done by a healthcare professional under national law and for preventive medicine, medical diagnosis or treatment or for the management of health-care services[42].

This is refined by the the Telemediengesetz. § 13, section (1) states, that the controller

---

36 cf. The European Parliament and the Council of the European Union (1995), Article 2, (d)

37 cf. The European Parliament and the Council of the European Union (1995), Article 4, 1., (a) and (b)

38 cf. The European Parliament and the Council of the European Union (1995), Article 4, 1., (c)

39 cf. The European Parliament and the Council of the European Union (1995), Article 7, (a)

40 The European Parliament and the Council of the European Union (1995), Article 8, 1.

41 cf. The European Parliament and the Council of the European Union (1995), Article 8, 2., (a)

42 cf. The European Parliament and the Council of the European Union (1995), Article 8, 3.

has to inform the user in a commonly understandable manner about the data which is collected and the form of processing of this data[43]. For a legal consent, the controller has to ensure, that the user is aware of his consent, that the consent is minuted, that the content of the consent is always available to the user and that the user can revoke his consent[44].
§§ 91, 93 and 94 of the Telekommunikationsgesetz states the same laws[45].
Also the data protection act of North Rhine-Westphalia constitutes the same laws[46] with the only restrictions, that its scope is limited to North Rhine-Westphalia.

Therefore ePill should explicitly inform the user that no data is stored and only anonymized transacted to find matching results, to comply with the stated laws.

### 4.1.2 Best Practices

The World Wide Web Consortium (W3C) has published a document in 2008 which states the basic best practices for developing for the mobile web. This document states 60 best practices, which shall ensure a minimum quality level for mobile web applications. These best practices emphasize the need of regard of the device's capabilities and supported technologies[47].
This document focuses on mobile web development[48], which has of course differences to native app development (e.g. the usage of frames and the accessibility of the device's specific features). Most of the best practices are applicable in both development environments.

For this specific project, which does not need more specific device capabilities, such as positioning and navigation features, we can focus on best practices related to the user interface, input and navigation methods as well as general best practices. Depending on the framework chosen, some of the best practices are already dealt with by the framework or

---

43 cf. Bundesregierung der Bundesrepublik Deutschland (2007), § 13, section (1)

44 cf. Bundesregierung der Bundesrepublik Deutschland (2007), § 13, section (2)

45 cf. Bundesregierung der Bundesrepublik Deutschland (1996), Section 2, §§ 91, 93, 94

46 cf. Der Innenminister des Landes Nordrhein-Westfalen (2000), Section 1, §§ 2, 4, 5

47 cf. World Wide Web Consortium (2008), e.g. 2., 11., 21., 42.

48 cf. World Wide Web Consortium (2008), Abstract

at least supported. I.e. a thematic consistency[49] is provided by native apps by default and by frameworks such as the TouchKit for Vaadin as well. Although they can be overridden, they provide a consistent theme. Wessels, Purvis, Rahman (2011) support the importance of a consistent appearance, also in comparison to a desktop application, if existent[50]. Lica (2010) further limits this to specific elements and points out, that mobile apps should provide just enough functionality to be useful and should not replicate the desktop optimized website[51].
Other best practices like utilizing a navigation bar at the page's top[52] for the main navigation have already become a standard across different platforms and frameworks.

Best practices which are mainly determined by implementations of the developer, like the usage of colors[53] or the chosen input methods[54] are often supported by the different platforms or frameworks but cannot be guaranteed by those. Even if different input methods like a number pad for numeric inputs are provided by the framework or platform they still need to be adapted and utilized by the developer to act in line with the best practices.

World Wide Web Consortium (2008) furthermore specifies a "Default Delivery Context"[55], which defines the minimal capabilities for mobile devices which should be supported. Tab. 4-1 illustrates the minimal capabilities suggested by W3C.
Nowadays it will be hard to match all of the requirements. E.g. a total maximum page weight of 20 kilobytes corresponds to the average file size of a 200 by 120 pixel JPEG-compressed file is about 10 kilobytes[56] and two images would already exceed the maximum page weight. With mobile devices like a Samsung Galaxy S3 which has a minimum of 720 pixel wide display, 120 pixels are far too less.

---

49 cf. World Wide Web Consortium (2008), 1.

50 cf. Wessels, Purvis, Rahman (2011), p. 2

51 cf. Lica (2010), p. 66

52 cf. World Wide Web Consortium (2008), 8.

53 cf. World Wide Web Consortium (2008), 26., 27

54 cf.World Wide Web Consortium (2008), 55., 56., 57.

55 cf. World Wide Web Consortium (2008), 3.7 Default Delivery Context

56 Tested with 60% compression rate and a random photograph

| **Parameter** | **Value** |
|---|---|
| Usable Screen Width | 120px |
| Markup Language Support | XHTML Basic 1.1 delivered with content type application/xhtml+xml. |
| Character Encoding | UTF-8 |
| Image Format Support | JPEG.<br>GIF 89a. |
| Maximum Total Page Weight | 20 kilobytes. |
| Colors | 256 Colors, minimum. |
| Style Sheet Support | CSS Level 1. In addition, CSS Level 2 @media rule together with the handheld and all media types. |
| HTTP | HTTP/1.0 or more recent. |
| Script | No support for client side scripting. |

Tab. 4-1: Default Delivery Context[58]

Also nearly any mobile browser supports client side scripting (e.g. JavaScript). For more detail, http://caniuse.com has compatibility lists of different browser features for nearly any browser. The parsing of JavaScript Object Notation (JSON) for example is supported by 93.41% of all mobile browsers[57].

Nevertheless, minimizing the total page size is still a concern. Wessels, Purvis, Rahman (2011) points out, that smaller pages lead to faster load times and therefor provide a more efficient experience for the user[59]. Nicolaou (2013) suggests different approaches to reduce page size as well as load time: Scripts and markup should be minified[60] and included inline[61] where it is possible. Preloading components and reducing DNS lookups can also result in a faster user experience[62].

Generally, Nicolaou (2013) recommends using a Content Delivery System (CDN), putting

---

57 cf. http://caniuse.com/#cats=JS_API, JSON parsing

58 cf. World Wide Web Consortium (2008), 3.7 Default Delivery Context

59 cf. Wessels, Purvis, Rahman (2011), p. 1

60 cf. Nicolaou (2013), p. 49

61 cf. Nicolaou (2013), p. 50

62 cf. Nicolaou (2013), pp. 48, 49

style sheets at the page's top and scripts at the bottom and using resized images rather than scaling them via HTML or CSS[63].

A study by Dahanayake et al. (2010) came to the result, that 71% of all responding web developers knew about the best practices, but only 11% said, that they understand these, 56% have a vague understanding and 33% do not understand the best practices [64].

Ayob, Nurul Zakiah binti, Hussin, Ab Razak Che, Dahlan (2009) adjusted and combined four different guidelines for application development, namely Shneiderman's Golden Rules of Interface Design, Seven Usability Guideline for Mobile Device (Abid Warsi, 2007), Human-Centred Design (ISO Standard 13407) and Mobile Web Best Practices 1.0 (W3C). From those guidelines, they developed the Three Layers Design Guideline for Mobile Application[65]. This guideline consists of three phases, which themselves represent different contexts, namely analysis (and the context of use), design (the context of medium) and testing (the context of evaluation). Tab. 4-2 illustrates this guideline.

This thesis will follow the Three Layers Design Guideline, as it is the latest guideline and combines multiple approved other guidelines. The third phase will likely be shortened due to the temporal restrictions for this thesis. The exact process we followed will be outlined in the following subsections 4.2 Analysis, 4.3 The Planning Process, 4.4 The Implementation Process and 4.5 Validation of the Mobile App and the experiences made will be discussed in section 5 Lessons Learned.

### 4.1.3 Internal Requirements

For developing a mobile frontend for ePill, it is important to us, that the main functionality of the web client is optimized but not reduced. Therefor a good user interface is indispensable. All functionality should be accessible easily and without confusion for the

---

63 cf. Nicolaou (2013), pp. 49, 50

64 cf. Dahanayake et al. (2010), p. 85

65 cf. Ayob, Nurul Zakiah binti, Hussin, Ab Razak Che, Dahlan (2009), p. 430

66 cf. Ayob, Nurul Zakiah binti, Hussin, Ab Razak Che, Dahlan (2009), p. 430, Table IV

| Phase | | Context of Use and Activities |
|---|---|---|
| 1 | Analysis | **Use**: Specify user and organizational requirements<br>1. Identify and document user's tasks<br>2. Identify and document organizational environment<br>3. Define the use of the system |
| 2 | Design | **Medium**: Produce design solution<br>1. Enable frequent users to use shortcuts<br>2. Offer informative feedback<br>3. Consistency<br>4. Reversal of actions<br>5. Error prevention and simple error handling<br>6. Reduce short-term memory load<br>7. Design for multiple and dynamic contexts<br>8. Design for small devices<br>9. Design for speed and recovery<br>10. Design for "top-down" interaction<br>11. Allow for personalization<br>12. Don't repeat the navigation on every page<br>13. Clearly distinguish selected items |
| 3 | Testing | **Evaluation**: Evaluate design against user requirements<br>1. Quick approach<br>2. Usability testing<br>3. Field studies<br>4. Predictive evaluation |

Tab. 4-2: Three Layers Design Guideline for Mobile Application[66]

user. Interactive elements like buttons should be visibly salient and have an immediate response to reduce the user's uncertainty. The general design, the color scheme and the fonts should be used in line with the web application to improve the recognition value. Another top priority is the accessibility of the app for as many users as possible. Therefor it is needed to provide a cross-platform app to be accessible for as many mobile platforms as possible and to have an intuitive user interface which also enables e.g. elderly people to use it efficiently.

Modularity and flexibility is another important factor. ePill is designed to be flexible

and scalable and the mobile client should incorporate the same idea. E.g. a scanning of barcodes on the packaging of pharmaceuticals could be implemented on a later stage to even further ease the use and increase the effectiveness.

## 4.2 Analysis

### 4.2.1 Assignment of a mHealth App Category

The mobile app does not differ from the web application in terms of privacy risks, content or connectivity. The mobile app aims to provide the same main functionality as the web application optimized for mobile devices. Therefor it also belongs to the same connectivity category, the medical connectivity, as the web application. Also no data is stored on the device and no additional information is sent to the server.
We plan to implement every request to the server to be optimized for SSL-encryption as soon as the server is capable of accepting and responding with SSL-encryption.

Therefor we would suggest to categorize the ePill mobile application as a low privacy risk, drug- and safety-related medical connectivity mHealth application and should be categorized similar to the web application.
Possible future features might change the classification (e.g. the addressed barcode scanning) if data handling or storage might be altered and therefor other privacy risks may arise.

### 4.2.2 The Different Operation Systems

#### 4.2.2.1 Android

Android is a mobile OS developed by the Open Handset Alliance[67], with Google being one of the biggest members. It is linux based and was unveiled in 2007. Android is released by Google under the Apache License and is therefor Open Source.[68] Developing for Android requires the Android SDK (or NDK). With the SDK developing apps is

---

67 http://www.openhandsetalliance.com

68 cf. http://source.android.com/source/licenses.html

done by writing Java code and writing the layout in specific XML[69]. Android apps are by default executed in the Dalvik managed runtime[70], except if they utilize the NDK. With the NDK apps can be (partly) written in C or C++ and are executed outside the Dalvik runtime[71].

While Android is adapted by many manufacturers and while it is also widely adapted by users, a software-based and a hardware-based fragmentation is clearly visible[72]. This fragmentation offers the user the choice to find exactly what he is looking for and enables more personalization. This fragmentation may lead to non-consistent applications on different devices as well as delays in updates. According to Google, an Android version released 2010 (2.3 "Gingerbread") has still a distribution of around 30.7%[73]. This also implies that developers do not only need to regard latest versions of the OS, but also older versions, which in return means that developers cannot always take full advantages of new capabilities as well as they need to pay much more attention to backwards compatibility.

For Android multiple IDEs are available. Eclipse[74] is one of the most popular and was one of the first, which supported the Android SDK. Android Studio, a for Android optimized version of IntelliJ IDEA[75], is still in development but already a stable IDE and greatly supported by Google[76].

Apps can be distributed directly or via an app store, like the Google Play Store[77]. The Google Play Store has some guidelines, which must be followed[78], but no review process

---

69 cf. for further details http://developer.android.com

70 cf. http://source.android.com/devices/tech/dalvik/index.html

71 cf. http://developer.android.com/tools/sdk/ndk/index.html

72 cf. for this and the next two following sentence Dan Han et al. (2012), pp. 83, 92

73 cf. http://developer.android.com/about/dashboards/index.html, visited 09/09/2013

74 http://www.eclipse.org

75 http://www.jetbrains.com/idea/

76 cf. http://developer.android.com/sdk/installing/studio.html

77 https://play.google.com/store

78 https://play.google.com/about/developer-content-policy.html

is performed.

#### 4.2.2.2 iOS

iOS is the proprietary OS developed by Apple for mobile devices. It was first introduced in 2008. In contrast to the Android OS only Apple develops hardware and software. According to Apple 94% of all active iOS devices run the latest version of iOS 6[79]. Compared to Android, all versions of iOS released before 2011 have a cumulative distribution of only 1%. Hardware-based fragmentation on iOS is mainly based on the screen size: Two different for the phones and one for the tablets.

iOS apps can only be developed on Xcode[80], which is only available for Apple Mac. Xcode combines user interface design and coding. Coding is mainly done by writing Objective-C code but also supports native code like C or C++. Designing interfaces is done by a user interface.

In contrast to Android, iOS apps can only be published via the Apple App Store[81], or on registered devices with a special license by Apple[82]. For submitting apps to the Apple App Store, one must obtain a developer license by Apple[83]. After submitting, all apps are reviewed and compared to Apple's guidelines[84].

#### 4.2.2.3 Windows Phone 7 and 8

Windows Phone 7 was released in 2010 as the successor of Windows Mobile. Windows Phone 8, the phone version of Windows 8, was released in 2012. Both utilize the "Metro" design, a tile-based design.

---

79 cf. for this and the first following sentence https://developer.apple.com/devcenter/ios/checklist/, last visited on 09/11/2013

80 https://developer.apple.com/xcode/

81 http://appstore.com

82 cf. https://developer.apple.com/programs/ios/enterprise/, last visited 09/11/2013

83 cf. https://developer.apple.com/programs/ios/, last visited 09/11/2013

84 https://developer.apple.com/appstore/guidelines.html

Development for Windows Phone requires Visual Studio[85] as IDE. Visual Studio is only available for Windows. C# or Visual Basic are the main programming languages and XAML is used for user interface design[86]. C++ can be utilized for graphic intensive applications.

The Windows Phone Store[87] is a closed store like the Apple App Store, and an enrollment is needed to publish apps. Other than the other stores, the Windows Phone Store offers the possibility to try apps out with reduced functionality. Also, the release of an app is preceded by a review process [88].

#### 4.2.2.4 other

Depending on the source for statistics, different OS are the respective market share leaders. Nevertheless other OS, e.g. Symbian, which was an important OS in 2008 with 47% market share of smartphone OS[89], is nowadays not listed at all or with less than 10% market share[90, 91, 92].
Therefore we will not take these OS into account, whose combined marketshare is around only 10%. This would require too much additional effort.

### 4.2.3 Possible Frameworks and Technologies

#### 4.2.3.1 Completely native

Building native apps for supporting Android, iOS and Windows Phone, means maintaining three different projects in three different programming languages and three different

---

85 http://www.microsoft.com/visualstudio/

86 cf. for this and the first following sentence http://msdn.microsoft.com/en-US/library/windowsphone/develop/ff402529(v=vs.105).aspx, last visited 09/11/2013

87 http://www.windowsphone.com/store

88 http://msdn.microsoft.com/en-us/library/windowsphone/develop/hh184843(v=vs.105).aspx

89 cf. "Canalys research release 2008/112" cited by Lin, Ye (2009), p. 622, Figure 1

90 http://gs.statcounter.com/#mobile_os-ww-yearly-2008-2013

91 http://www.idc.com/getdoc.jsp?containerId=prUS24257413

92 http://blogs.strategyanalytics.com/WSS/post/2013/08/01/Strategy-Analytics-Android-Captures-Record-80-Percent-Share-of-Global-Smartphone-Shipments-in-Q2-2013.aspx, Exhibit 1

user interface definitions. But native apps offer the most seamless user interface integration into the OS and the best performance.
While the seamless user interface integration could help people using the app in a familiar context, performance is not an issue for the ePill project. The costs for learning the specific aspects of those frameworks, developing and maintaining the three implementations are not reasonable for the given time frame of this thesis.

Additionally, we do not have an existing web service which the app could utilize to receive data from the server. The current utilized framework Vaadin does not offer an easy way to provide a web service with the logic already implemented. This additional effort is definitely a decisive argument.

#### 4.2.3.2 HTML 5, jQuery mobile and Phone Gap

Providing an app for nearly any mobile device is possible with a web app. Based on web technologies like HTML, CSS and JavaScript, apps can be brought to nearly any mobile OS with only one implementation.
As most mobile devices support HTML 5[93] and JavaScript, frameworks like jQuery mobile[94] provide a common looking user interface without much additional effort. PhoneGap[95] enables web-based apps to access the device's capabilities like the camera or local storage[96].

Still, with this approach we would need a web service, which we could consume with jQuery. The web app approach lessens the effort by developing only one app but still has the time consuming need for a new web service.

#### 4.2.3.3 Xamarin

---

93 http://www.w3.org/TR/2012/WD-html51-20121217/

94 http://jquerymobile.com

95 http://phonegap.com/

96 for a detailed overview: http://phonegap.com/about/feature/

Xamarin[97], also known as MonoTouch, is an IDE and framework, which produces native apps for Android, iOS and Windows Phone from just a single C# code basis. Different user interface definitions are still needed, but it combines the advantages of native apps and web apps.
Xamarin utilizes C# and can be used with Visual Studio or Xamarin Studio. The later is available for both Windows and Mac OS X.

But also apps developed with Xamarin would need an additional web service, which we cannot provide in the given time. Additionally, Xamarin is only with specific requirements, so probably it would be more expensive than providing a web app.

#### 4.2.3.4 Vaadin and TouchKit

Vaadin[98] is a Java framework, with which the developer does only need to write code and define a user interface. The framework builds the user interface and handles all communication between the client and the server. TouchKit[99] enables a Vaadin project to easily add a mobile web client. It provides various controls which were optimized for mobile devices. It also provides access to the device's capabilities, like positioning, offline storage or camera.

TouchKit supports iOS 5 or newer, Android 2.3 or newer and Windows Phone 8. Vaadin is focused on WebKit[100]-based browsers[101], still it is compatible to most of the mobile browsers.

Using Vaadin and TouchKit for ePill has the great advantage, that we would not need an additional web service, we could use the existing code and just add a TouchKit-based user interface, which seamlessly adapts to the existing code.

---

97 http://xamarin.com

98 https://vaadin.com

99 https://vaadin.com/touchkit

100 http://webkit.org

101 cf. https://vaadin.com/book/-/page/mobile.considerations.html, last visited 09/11/2013

### 4.2.4 The Choice for Vaadin and TouchKit

Finally we chose Vaadin and the TouchKit Add-On as framework for the mobile frontend. The main reason is the lack of an easy accessible web service in Vaadin itself. Without a web service, we would first have had to build a web service to have a connection from the mobile frontend to the database and the application's logic. Independently from the framework chosen for the frontend this would have been a large additional effort which we could not have completed before the end of this thesis.

Furthermore we wanted the complete system to be as homogenous as possible. The web application uses Vaadin as main framework. Utilizing the TouchKit Add-On for Vaadin, we reused as much existing code as possible and infrastructure by only adding another layer. This results in a much improved maintainability as the coding style is the same for the web application and the mobile application. Additionally no additional IDEs or frameworks need to be included or maintained.

## 4.3 The Planning Process

As already mentioned, the ePill system is already a functional prototype and the yet to develop mobile frontend is mostly an additional user-interface. Therefor no system-wide planning is needed, only data handling and user interfaces have to be planned.

Having the already existing application logic in mind, the planning of data handling is reduced to a functional planning. To be able to have the functions available at the right time for the user, we have to combine the planning of the functional aspects with the user interface. Therefor we will plan the user's flow through the application (user flow), which pays attention to the accessibility of the functions in the right context, and hence combines functional and user interface planning. For the ease of explanation, we added some mockups and screenshots. We decided to print them uncolored to prevent distraction from highlighting colors and because the color scheme is not final yet. We decided to focus on smaller mobile devices like phones and not on tablets, because smaller user interfaces can be expanded and are still usable but the reverse is not always true. We will optimize the app's implementation for different layout depending on the device's screen size, but probably we will not implement different layouts during this thesis' timeframe.

The three major functions of ePill (Search, Display and Supplementing Services) should be accessible as fast as possible. Of course, the display functionality and some supplementing services (e.g. the term explanation functionality) are only able to present results, if a pharmaceutical is selected. So we designed Fig. 4-1 as a stripped down starting screen with every main function quickly available. Due to some missing generic controls provided by Vaadin TouchKit, like the search bar, we finally implemented the start screen as illustrated in Fig. 4-2. This resulted furthermore in a more separated presentation of search string input view and result view, as presented in Fig. 4-3 and Fig. 4-4.

Throughout the planning and designing we tried to reproduce common patterns and user controls. Therefor we utilized the navigation back on the top left corner, illustrated by a leftwards oriented arrow-shaped button. Central navigation which specified the view, e.g. main function selection or the selection of a specific pharmaceutical from e.g. a search result list, is centered in the main content area in the screen's center.

Most of the views offer a toolbar at the screen's bottom. This toolbar contains controls which are used for navigation inside the screen's center view, e.g. go back to top or navigate to a specific header. Especially in the pharmaceutical details view with much information visible, it can be very handy to jump right to the header one is interested in. Additionally, some controls are added in the toolbar which enable further interaction with the information displayed, e.g. adding the currently visible pharmaceutical to the comparison list, which I will discuss later.

The web application has rich customization possibilities. One can adjust the font size, the details of pharmaceuticals to be displayed and the layout of the web application itself. As it turned out, it is not applicable for the user to change the mobile apps layout, because the user interface so too compact to add additional elements and too few elements are visible to hide any element.

The comparison list is a concept added to the mobile application and derived from a view in the web application. The web application offers a view in which pharmaceuticals


Fig. 4-1: Main Screen Mockup

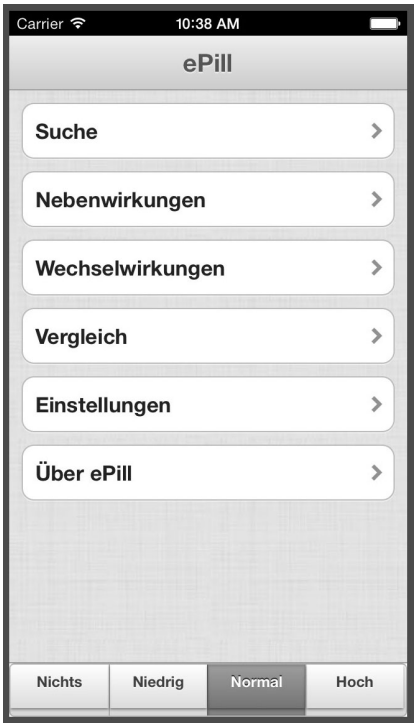


Fig. 4-2: Final Main Screen


Fig. 4-3: Search Input Screen

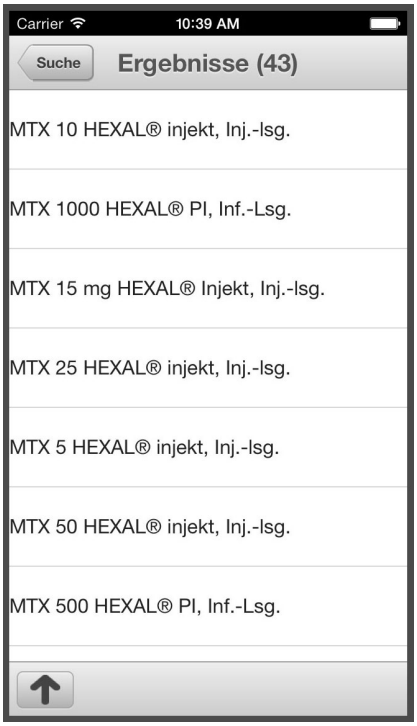


Fig. 4-4: Search Result Screen

can be added and afterwards, via a button click, aggregated information, like adverse reactions, can be listed. Because we have much less available screen space in the mobile app, we decided to have the list globally available throughout the entire app. E.g. if one searched for a pharmaceutical and has selected one to display detailed information, one can easily via a button in the toolbar add this pharmaceutical to one's comparison list. If one now selects e.g. the side effects functionality in the main screen, one sees his comparison list and can add more to it. This screen is illustrated in Fig. 4-7. If one decides to do so, one is presented a search input screen but can in the search result screen only tick on those pharmaceuticals one wants to add to one's comparison list and not see the pharmaceutical details on click. This is illustrated in Fig. 4-8.

Fig. 4-7 also highlights the toolbar. The leftmost button scrolls back to the top, whereas on the right side, we have on the one hand an add button, which brings up the already discussed search view and the trash button, which clears the comparison list.
The button on the top right corner labeled "Go" starts the currently selected function, e.g. adverse reactions on the comparison list.

To prevent users from faulty operations, we will add dialogs asking to confirm the chosen operation where adequate. This currently only fits to the delete and add operations of the comparison list, as illustrated in Fig. 4-11.
Fig. 4-12 illustrates the pharmaceuticals details screen. This screen presents all information for a specific pharmaceutical, e.g. after the selection of one pharmaceutical in the search results screen or from the comparison list as illustrated in Fig. 4-7. Basically, this screen consists of the back navigation button in the top left corner, the toolbar in the bottom and the centered main content area as most of the mobile app's screens. While the main content area displays the leaflet information segmented and only those informations as chosen, the toolbar offers different operations. On the lower right corner, the add button allows the user to add this specific pharmaceutical to his comparison list. The left side of the toolbar offers on the one hand a back to top button just like the comparison list screen and on the other hand a navigation button, enabling the user to jump to any of the sections by which the main content is segmented. A click on this button reveals a list of the sections through which the user can scroll and select one to jump to. Another click on

this button closes the list as well as the selection of a section.

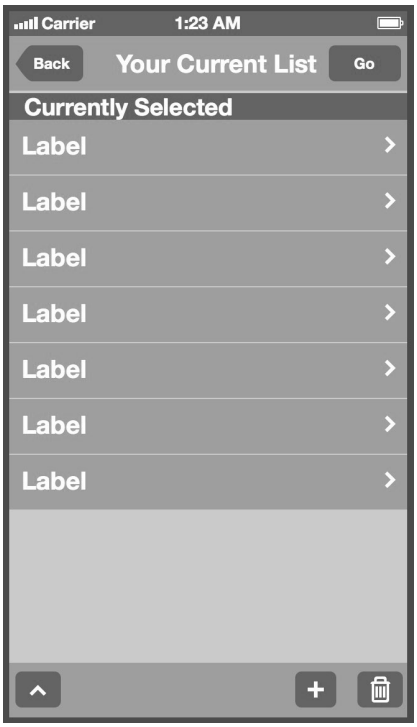


Fig. 4-5: Comparison List Mockup

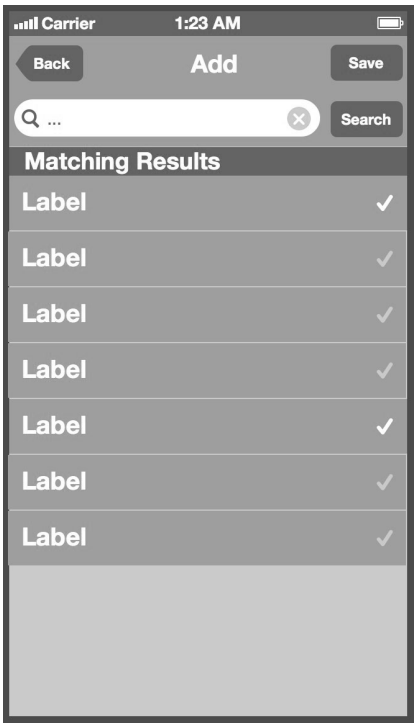


Fig. 4-6: Results Screen Mockup

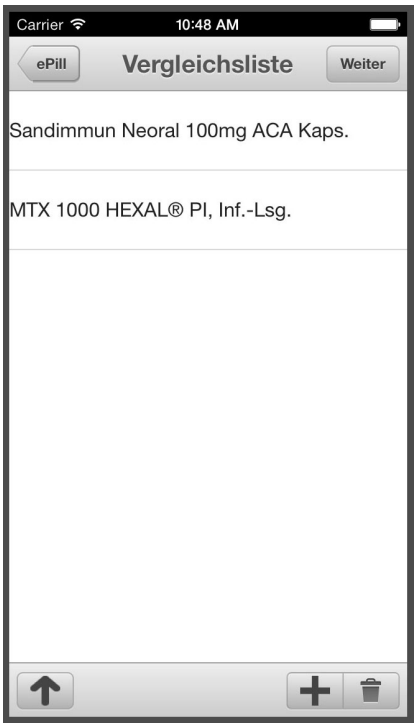


Fig. 4-7: Comparison List Screen

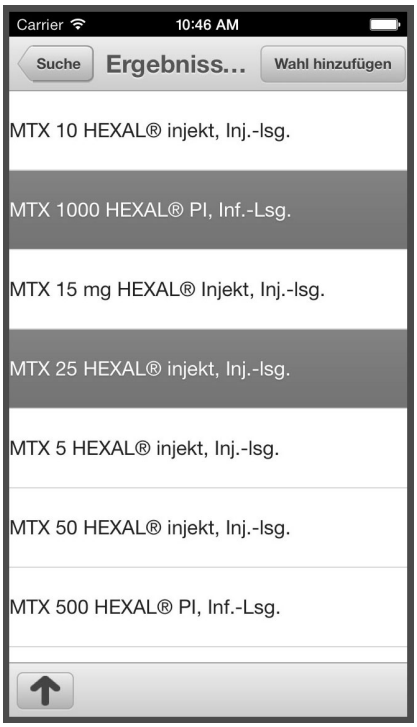


Fig. 4-8: Results Screen

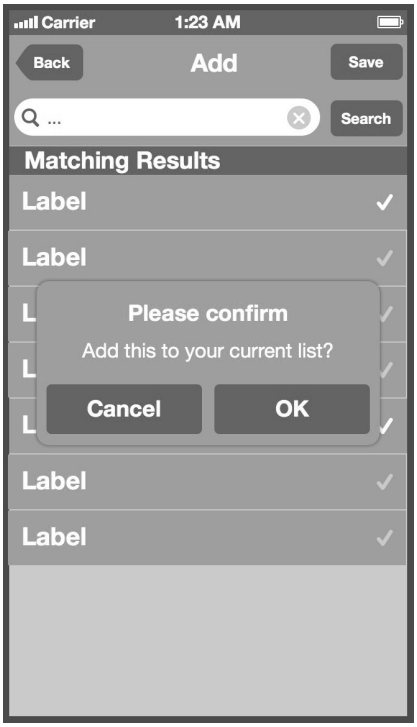


Fig. 4-9: Confirm Dialog Mockup

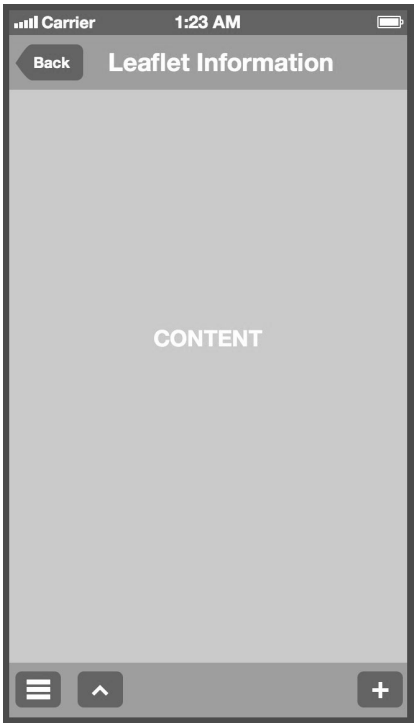


Fig. 4-10: Details Screen Mockup

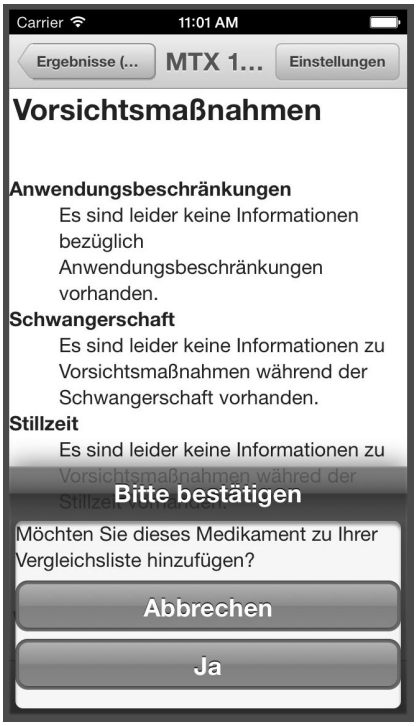


Fig. 4-11: Confirm Dialog

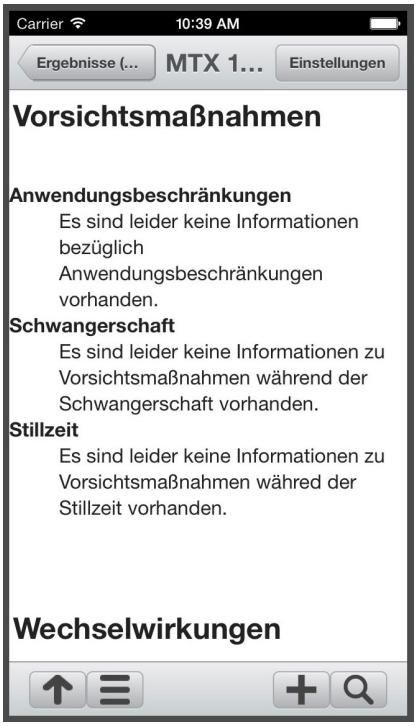


Fig. 4-12: Details Screen

## 4.4 The Implementation Process

The implementation of the planned user flow was struggling at the beginning because of some differences of Vaadin and TouchKit to other frameworks we worked with beforehand. E.g. TouchKit does not provide a search bar which can be attached to the navigation bar as planned in the mockups. Furthermore the ePill web application is currently based in Vaadin 6, whereas TouchKit 3 is based on Vaadin 7. We tried to utilize Vaadin 6 and TouchKit 2 for the mobile app but as it turned out, TouchKit 2 has many drawbacks in comparison to TouchKit 3, like far less documentation, tutorials and example projects. The choice for Vaadin 7 and TouchKit 3 had the result, that we could not just include the ePill web application and directly access the business logic, as some adjustments were needed to upgrade a Vaadin 6 project.
Because time was still a matter to us, we decided to include as much as possible from the web application and copy and modify as less as possible. During further development, when the web application is upgraded to Vaadin 7, both projects can be merged together and code redundancies can be removed. The choice for Vaadin 7 brought some additional work for us as we had to adjust references to e.g. the main window and application because their handling did change.

After having set the stage for the implementation, we discovered that Vaadin does not seem to have a community as big and as active as e.g. Node.js[102] or django[103]. While we made the experience that to most issues an answer can be found in the related forums, mailing lists or community pages as well as on more general websites like Stack Overflow[104], only very few answers and questions were found on Stack Overflow regarding Vaadin or TouchKit. A direct comparison between the official django mailing list[105] and the Vaadin forum[106] revealed, that while the django mailing list had 40,367 topics[107], the

---

[102] http://nodejs.org

[103] https://djangoproject.com

[104] http://stackoverflow.com

[105] https://groups.google.com/forum/#!forum/django-users

[106] http://vaadin.com/forum

[107] last visited on 09/16/2013

Vaadin forum had 14,523 threads[108], including news and forum guidelines. Nevertheless, the Vaadin forum was often helpful.

Programming with Vaadin is easy after the first steps into the framework are succeeded. The coding style supposed by the framework was consistent and helped achieving fast results. Nevertheless, missing controls are a drawback and the inflexibility of the framework is both an advantage and disadvantage.

The lack of components like alert- and confirm-dialogs or search bars is compensable. During the development, we utilized popovers to imitate the well known appearance of dialogs in most of the mobile OS. Vaadin's popover component is functional, as it provides an overlay with a responsive layout for different screen sizes but is not completely customizable, e.g. the bar at the popover's top cannot be customized and buttons cannot be added to this bar. Vaadin's popover has also a much bigger border than we would have used with a common dialog component. This reduces the useable content area. Additionally the borders distract the user and reduces the focus on the content and buttons.

We implemented a workaround for the missing search bar attached to the navigation bar by adding an additional view. As stated beforehand, we planed to attach the search bar directly beneath the navigation bar to enable the user to quickly access the search functionality. Results were planned to be displayed as an overlaid list. We had to abandon the idea of an overlaid list as well because the Vaadin TouchKit framework did not have the needed functionality at hand. Overlays looked either unfamiliar or complicated the use of the app, especially for dismissing the results. Finally we chose to display a view containing an input field, a button and a label displaying information like the minimum length of the query. The results are displayed in another standalone view as plain list.

Another issue we discovered was the drop for dynamic style sheets and themes in Vaadin 7. In Vaadin 6 one could swap the used theme during runtime, which was used to adjust the font size. In the time given we could not implement a workaround. We tried by adding

---

[108] last visited on 09/16/2013

CSS classes to the different controls but this either meant a high complexity for the algorithm to discover all controls or a high need of memory if we were to keep references for the controls.

The Vaadin framework did help to get fast results. E.g. the settings screen for customizing the displayed leaflet information could be implemented in a short time thanks to the automatic handling of references between the client- and the server-side. Nevertheless it was not possible to receive specific events like a "will appear" or "did appear" for views. These events are needed for refreshing the view or specific controls if needed. Java and Vaadin offer listeners for "value changed" events but those still need a reference for the controls, which have to be passed along through the views. The main screen contains a toolbar at the screen's bottom to select a preset of visible information on leaflets. If now the user decides to change the preset in a lower view, this toolbar needs to be updated as well. Additionally to the increased reference count this method has the drawback that the toolbar gets updated on every value change, although it is not visible. With a "will appear" event the update would have been done only once directly before the toolbar becomes visible.

Much of the layout from the web application could be reused with only minor adjustments, thanks to Vaadin's seamless TouchKit integration. The comparison functionality is still an issue. Comparing two pharmaceuticals can only be effective, if related information is presented closely. During the development we experimented with a vertical instead a horizontal layout, meaning that each related information of the two pharmaceuticals are presented beneath each other, not side by side, but it became very confusing because of the doubled page length. For now we concluded that a horizontal layout and a landscape orientation will be best in this case.

## 4.5 Validation of the Mobile App

In this section we will compare the developed app to the norms and best practices highlighted in section 4.1 Preconditions, as well as to all requirements stated in section 4.2 Analysis.

Because no user-related data is stored, neither personal nor usage related, we do not have any conflicts with the norms mentioned in section 4.1.1 Norms for Mobile Apps.

As already stated, Vaadin and TouchKit support developers by providing a default theme for applications to maintain a consistent user interface, at least design and color wise. Despite the statement of Wessels, Purvis, Rahman (2011) that a consistency to a desktop application should be maintained[109], we focused on an easy to use user interface on mobile devices. Some parts look very familiar on both applications, e.g. the detailed informations or the comparison of pharmaceuticals, but the navigation is absolutely different as a result of the limited screen size and a touch- instead of mouse-based operation. We chose this differentiation with the statement of Lica (2010) in mind that the mobile app should only provide enough functionality to be useful[110]. This resulted likewise in a missing customizable tab-bar or different sidebars. These elements might be operational on a bigger screened device such as tablets, but is not useable on most mobile phones. Further development of the mobile app might develop a optimized tablet user interface and integrate this functionality.

In sections 4.3 The Planning Process and 4.4 The Implementation Process we stated that we utilized a navigation bar and a toolbar as main interaction controls for navigation and direct interaction such as clearing lists. These two concepts of placing controls on the edges of the screen are also recommended by World Wide Web Consortium (2008)[111].

We also tried to follow common input methods. Because we did not have specific input fields for e.g. numbers, the only specialization we could use was the enter-key equivalent on mobile devices, often a button on the lower right corner of the simulated on-screen keyboard.

World Wide Web Consortium (2008) suggested a "Default Delivery Context"[112] illustrated in Tab. 4-1 Mobile Default Delivery Context. Vaadin does not leave much to the developer to optimize the rendering of the controls as well as the organization of requests

---

109 cf. Wessels, Purvis, Rahman (2011), p. 2

110 cf. Lica (2010), p. 66

111 cf. World Wide Web Consortium (2008), 8.

112 cf. World Health Organization (2011), 3.7 Default Delivery Context

and client-side handling of data. Developers can only optimize used images and CSS style sheets. Because only very few images are used and not much custom styling is applied, we could not pursue this recommendation in particular.

Comparing this app to the Three Layers Design Guideline developed by Ayob, Nurul Zakiah binti, Hussin, Ab Razak Che, Dahlan (2009), illustrated in Tab. 4-2 Three Layers Design Guideline for Mobile Application, results in the following: While phase one, the analysis, was already done by Dehling, Sunyaev (2012a), phase two was one of the purposes of this thesis. We tried to implement shortcuts where possible with the help of toolbars and quick access to settings, specially in the detail view, and offered where possible informative feedback, e.g. for the search right beneath the input field or in popovers for detailed information. Using the same theme and general user interface layout led to the proposed consistency as well as the optimization for small devices. The utilization of a navigation bar at the top also compiles with the proposed "top-down" interaction or navigation. Vaadin as framework as well as the already existing code support error prevention or simple handling of those. Personalization is made possible with the settings view which enables the user to select only those information to be displayed he is interested in.
Phase three, testing, could only be done superficially until the end of this thesis' time frame. Further development might pursue a more detailed testing of the mobile app.

The internal requirements were compiled with mostly, buttons were made salient and interaction with them resulted in immediate feedback, e.g. the search button disables itself until the server processed the request hence the user cannot execute the search twice. The mobile app is available on nearly any mobile platform thanks to the HTML 5 standard and its good distribution among current browsers. Also, the app is as modular as the web client because much code was reused and mostly only adapted to the changes in Vaadin 7 compared to Vaadin 6 and the layout which was optimized for small screens. Only the color scheme was not adapted because of the already addressed more general familiarity among mobile OS with the default theme.

Validating the app from an overall point of view, we propose that further development

of the client may lead to an even more effective use for the user with a focus on an optimization in the details, especially for the basic theme. We would set the focus on the popovers which are currently not matching perfectly to the overall appearance. Nevertheless, the mobile app follows the guidelines stated in section 4.1 Preconditions and should provide a good user interface and functionality. A dedicated usability test may reveal some remaining weak points.

# 5. Lessons Learned

We learned very different lessons during development. mHealth apps do have some similar requirements as any app, but have to deal with much stronger information security concerns compared to casual apps. They also need to pay more attention to usability and availability than casual apps by definition.
In the case of ePill information security is not an issue because no user related information is stored and with the web-based mobile app ePill is available on nearly any mobile device. But developing with Vaadin as framework was not always a good choice. Vaadin simplified the development process but slowed down at the very beginning and had a price to pay: The app is not as perfectly designed as it could have been with more influence on the client-side, especially on the user interface in terms of available controls.

Starting with Vaadin is not really hard but takes some time to getting started. We had some trouble getting started with Vaadin's installation because our installed version of Eclipse IDE[113] was not compatible, the same happened to our installed version of Maven. After having read the Vaadin's Beginner Guide[114] we only had minor issues understanding Vaadins architecture and development went mostly quick and easy. As already stated, some missing controls were a drawback to us and workarounds had to be found, but all in all it is a good solution for a quick development without the need of further knowledge in HTML, CSS or JavaScript. Having not to deal with cross-browser-optimizations was a relieve.
After the development of the mobile app we would nevertheless propose a different approach. If Vaadin is already utilized in the existing system, it is good to reuse the code. But if not, or if a web service is available, we would propose native applications, maybe developed with Xamarin[115]. This offers the possibility to fine tune the user interface much more, offer a much more familiar look on the different OS and much more controls are available. It is worth the additional effort of developing a web service and different user interfaces, at least their OS-specific definition for the better accessibility and extended

---

113 http://eclipse.org

114 https://vaadin.com/tutorial

115 http://xamarin.com

controls. Additionally, OS like iOS offer native apps the possibility to run in background and perform specific tasks energy efficiently, while web-based apps are forced to quit on exit. This constraint does not affect ePill but e.g. monitoring apps need a continuing execution.
Frameworks such as PhoneGap[116] provide accessibility to extended functionality of the device for most mobile OS but still do not offer background execution.

We furthermore learned that planning plays an important role for a fast and successful implementation. But more important is that the framework and it's possibilities is well known. During planning we had superficial knowledge about Vaadin and TouchKit, we knew some sample code and user interface, which made us believe that any controls from native applications can be utilized as well. If we knew during development that many controls are not implemented already our planned user interface would have looked different and we would have saved some time during the implementation.

**TODO: ONE MORE PARAGRAPH?**

[116] http://phonegap.com

# 6. Conclusion

**Bibliography**

Ayob, Nurul Zakiah binti, Hussin, Ab Razak Che, Dahlan (2009)

Ayob, Nurul Zakiah binti, Hussin, Ab Razak Che, Halina Mohamed Dahlan: “Three Layers Design Guideline for Mobile Application”. In: *Information Management and Engineering, International Conference on*. 2009, pp. 427–431

Badashian et al. (2008)

Ali Sajedi Badashian, Mehregan Mahdavi, Amir Pourshirmohammadi, Minoo Monajjemi nejad: “Fundamental Usability Guidelines for User Interface Design”. In: *Computational Sciences and Its Applications, 2008. ICCSA ’08. International Conference on*. 2008, pp. 106–113

Bundesregierung der Bundesrepublik Deutschland (1996)

Bundesregierung der Bundesrepublik Deutschland: Telekommunikationsgesetz: TKG, 1996. http://www.gesetze-im-internet.de/tkg_2004/, visited on 09/07/2013

Bundesregierung der Bundesrepublik Deutschland (2007)

Bundesregierung der Bundesrepublik Deutschland: Telemediengesetz: TMG, 2007. http://www.gesetze-im-internet.de/tmg/, visited on 09/07/2013

Dahanayake et al. (2010)

Ajantha Dahanayake, Caroline Collier, Daniel Glenzer, Tanya Goette, Richard Welke: Mobile Website Engineering and Mobile Web Best Practices Guidelines: A Reality Check. In: Journal of International Technology and Information Management. Nr. 2, Jg. 19, 2010, pp. 79–IV

Dan Han et al. (2012)

Dan Han, Chenlei Zhang, Xiaochao Fan, Abram Hindle, Kenny Wong, Eleni Stroulia: Understanding Android Fragmentation with Topic Analysis of Vendor-Specific Bugs. In: Reverse Engineering, Working Conference on. 2012, pp. 83–92

Dehling, Sunyaev (2012)

Tobias Dehling, Ali Sunyaev: Architecture and Design of a Patient-Friendly eHealth Web Application: Patient Information Leaflets and Supplementary Services. In: AMCIS 2012 Proceedings. 2012, pp. 1–8

Dehling, Sunyaev (2012)

Tobias Dehling, Ali Sunyaev: Improved Medication Compliance through Health IT: Design and Mixed Methods Evaluation of the Application ePill: Research-in-Progress. In: Thirty Fourth International Conference on Information Systems, Milan. 2012, pp. 1–11

Dehling, Sunyaev (2013)

Tobias Dehling, Ali Sunyaev: Information Security and Privacy Implications of mHealth Apps: An Overview. 2013, pp. 1–12

Der Innenminister des Landes Nordrhein-Westfalen (2000)

Der Innenminister des Landes Nordrhein-Westfalen: Gesetz zum Schutz personenbezogener Daten: DSG NRW, 2000. https://recht.nrw.de/lmi/owa/br_bes_text?anw_nr=2%5C&gld_nr=2%5C&ugl_nr=20061%5C&bes_id=4908%5C&aufgehoben=N%5C&menu=1%5C&sg=0, visited on 09/08/2013

d'Heureuse et al. (2012)

Nico d'Heureuse, Felipe Huici, Mayutan Arumaithurai, Mohamed Ahmed, Konstantina Papagiannaki, Saverio Niccolini: What's app?: a wide-scale measurement study of smart phone markets. In: SIGMOBILE Mob. Comput. Commun. Rev. Nr. 2, Jg. 16, 2012, pp. 16–27

Future of Privacy Forum, Center for Democracy & Technology (2011)

Future of Privacy Forum, Center for Democracy & Technology: Best Practices for Mobile Application Developers: App Privacy Guidelines. In: Future of Privacy Forum and the Center for Democracy & Technology. 2011, pp. 1–20

Istepanian, Jovanov, Zhang (2004)

R.S.H. Istepanian, E. Jovanov, Y.T. Zhang: Guest Editorial Introduction to the Special Section on M-Health: Beyond Seamless Mobility and Global Wireless Health-Care Connectivity. In: IEEE Transactions on Information Technology in Biomedicine. Nr. 4, Jg. 8, 2004, pp. 405–414

Kaletsch, Sunyaev (2011)

Alexander Kaletsch, Ali Sunyaev: Privacy Engineering: Personal Health Records in Cloud Computing Environments. In: ICIS 2011 Proceedings. 2011, pp. 1–11

Lica (2010)

Liviu Lica: Mobile and Social: Ten Best Practices for Designing Mobile Applications. In: Informatica Economica. Nr. 3, Jg. 14, 2010, pp. 60–74

Lin, Ye (2009)

Feida Lin, Weiguo Ye: "Operating System Battle in the Ecosystem of Smartphone Industry". In: *Information Engineering and Electronic Commerce, International Symposium on*. 2009, pp. 617–621

Martínez-Pérez, de la Torre-Díez, Isabel, López-Coronado (2013)

Borja Martínez-Pérez, de la Torre-Díez, Isabel, Miguel López-Coronado: Mobile health applications for the most prevalent conditions by the World Health Organization: review and analysis. In: Journal of medical Internet research. Nr. 6, Jg. 15, 2013, e120

Nicolaou (2013)

Alex Nicolaou: Best Practices on the Move: Building Web Apps for Mobile Devices. In: Communications of the ACM. Nr. 8, Jg. 56, 2013, pp. 45–51

Njie (2013)

C.M.L. Njie: Technical Analysis of the Data Practices and Privacy Risks of 43 Popular Mobile Health and Fitness Applications. In: Privacy Rights Clearinghouse. 2013, pp. 1–31

The European Parliament and the Council of the European Union (1995)

The European Parliament and the Council of the European Union: Directive 95/46/EC, 1995. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:NOT, visited on 09/08/2013

Wessels, Purvis, Rahman (2011)

Andrew Wessels, Mike Purvis, Syed Rahman: "Usability of Web Interfaces on Mobile Devices". In: *Information Technology: New Generations, Third International Conference on*. 2011, pp. 1066–1067

West et al. (2012)

Joshua H. West, P. Cougar Hall, Carl L. Hanson, Michael D. Barnes, Christophe Giraud-Carrier, James Barrett: There's an App for That: Content Analysis of Paid Health and Fitness Apps. In: Journal of medical Internet research. Nr. 3, Jg. 14, 2012, pp. 1–11

World Health Organization (2011)

World Health Organization: mHealth: New horizons for health through mobile technologies, 2011. http://whqlibdoc.who.int/publications/2011/9789241564250_eng.pdf, visited on 08/30/2013

World Wide Web Consortium (2008)

World Wide Web Consortium: Mobile Web Best Practices 1.0: Basic Guidelines, W3C Recommendation, 2008. http://www.w3.org/TR/2008/REC-mobile-bp-20080729/, visited on 09/09/2013

Yeh, Fontenelle (2012)

Shea-Tinn Yeh, Cathalina Fontenelle: Usability study of a mobile website: the Health Sciences Library, University of Colorado Anschutz Medical Campus, experience. In: Journal of the Medical Library Association. Nr. 1, Jg. 100, 2012, pp. 64–68

**Erklärung**

Hiermit versichere ich an Eides Statt, dass ich die vorliegende Arbeit selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

Köln, den 26. September 2013

## Curriculum Vitae

### Persönliche Angaben

| | |
|---|---|
| Name: | Phil Diegmann |
| Anschrift: | Wipperfürther Str. 477, 51515 Kürten |
| Geburtsdatum: | 06.02.1991 |
| Geburtsort: | Wipperfürth |
| Familienstand: | ledig |

### Schulische Ausbildung

| | |
|---|---|
| 09/1998 - 07/2002 | St. Antonius Grundschule in Wipperfürth |
| 09/2002 - 07/2010 | Engelbert-von-Berg Gymnasium in Wipperfürth, Abschluss: Abitur (1,5) |

### Studium

| | |
|---|---|
| 10/2010 - 09/2013 | Universität zu Köln, Wirtschaftsinformatik, B.Sc. |
| 10/2013 - 09/2015 | Universität zu Köln, Information Systems, M.Sc. |

### Praktika und Berufserfahrung

| | |
|---|---|
| 02/2007 | Krüger Industrieautomation GmbH, Wipperfürth (Praktikum) |
| 04/2008 - 02/2011 | Webergy Internet Software AG, Lindlar (Teilzeit) |
| 02/2012 - 02/2014 | Forschungsgruppe Informationssysteme und Lernprozesse, Universität zu Köln (Studentische Hilfskraft) |
| seit 08/2012 | Selbstständig (IT-Beratung, Entwicklung und Design) |

### Sonstige Qualifikationen und Auszeichnungen

| | | |
|---|---|---|
| Sprachkenntnisse | Deutsch: | Muttersprache |
| | Englisch: | Fließend |
| | Französisch: | Gute Kenntnisse |
| | Spanisch: | Grundkenntnisse |
| seit 10/2010 | Stipendiat der Studienstiftung des Deutschen Volkes | |
| seit 06/2012 | Sitz im Ausschuss "Schule, Generationen und Soziales" der Gemeinde Kürten | |