UNIVERSITÄT PADERBORN

Die Universität der Informationsgesellschaft

Faculty of Electrical Engineering,
Computer Science and Mathematics
Department of Computer Science
Database and Information Systems

Master Thesis

by

Jörg Amelunxen
Paulinenstrasse 9
33098 Paderborn
Student Registration Number: 650 44 65

Paderborn, October 16, 2014

DEVELOPMENT OF THE HIP-APPLICATION– NO SUBTITLE

Supervisor_____
Dr. Simon Oberthür

Examiners_____
Dr. Simon Oberthür
Prof. Dr. Gregor Engels

ABSTRACT

Lorem

ZUSAMMENFASSUNG

Ipsum

ACKNOWLEDGMENT

At this point I would like to thank all people who supported me and made this thesis possible in the first place.

Furthermore, I would like to thank all people who have participated at the survey, which was needed to evaluate the meaningfulness of the diagrams, and all people who proofread my thesis.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ABBREVIATIONS

AR	Augmented Reality

1

INTRODUCTION IN THE THESIS

1.1 WHAT IS THE CURRENT SITUATION WITHOUT THE TOOL/APP

At the present point in time, guests of the city Paderborn has to look up information about the city in a tedious way, for example using Wikipedia or other existing platforms. On the other hand, people, who want to provide information about the city (e.g., university employees or students), have to provide these information in a general accessible way, again like Wikipedia. Thus, they are limited to the features that are provided by these platforms. Since Wikipedia has been founded in 2004 by the Wikimedia foundation (Wikimedia-Foundation (2014)), most of the used technologies of the web application are nowadays outdated and very general. So, there is a rising need for a new technological updated approach, which is more focused on the specific topic of the city Paderborn and its history.

Especially the use of mobile devices has been risen in this time, which is easily recognizable at the number of sales of the Apple iPhone. The iPhone has been sold 0.27 million times in Q3 2007 and 51.03 million times in Q1 2014 (Statista (2014). Of course, this shift from the device side (i.e., hardware) includes a major shift in (software-) technology as well. Technologies like the nowadays well known AR would not be possible in 2004. Of course, this new technology includes a lot of new opportunities to transfer knowledge between people and cultures. AR is a great example to show how 'the real world' is blended more and more with artificial information; for example in the form of call-outs and layers. AR is a technology that displays virtual (i.e., digital) information on top of a real object or location using the camera of a mobile device as input for the real objects. So, it ends up to be a combination of both worlds; the real and the digital one. Azuma et. al. has shown a lot of possible fields where the usage of AR would be a great improvement, which includes the field of annotation

and visualization (Azuma (1997)). Furthermore, path finding and navigation are fields that could be revolutionized by using AR on mobile devices.

With a simple information website or app like Wikipedia, we include the tedious situation that the person that wants to get to the place he just read about, needs to input the address into another app to navigate him to the position. After he have arrived, he need to switch back to, for example, Wikipedia to manual compare the written information with the object or place he sees in front of him. If the person wants to change the shown information on his mobile device, he does this in general by using the touchscreen of the device. Nevertheless, he is comparing and looking at something that is placed in front of him. This leads to a break within the action and perception space (Hampel (2001)) and is a bad example with respect to the locality of the information (Bondo (2010)). As we will see, AR is a tool that we can use to remove this problem and join the action and perception space while keeping the locality of the information in mind. Furthermore, at the moment we have a lot of unnecessary overhead due to the needed app switching between the information app and the navigation app.

Now, even if somebody wants to publish information about Paderborn on Wikipedia to enable guests of the city to get knowledge about the environment, it is only possible to publish the information as static content (that includes text, graphics, audio and video). On top of that, it is not possible to review the information privately and in enough detail to create university courses that do not include a written paper as the final exam but an entry within such an information system. So, if we would have private annotations within a system that is owned by the university, it would enable the university employees to offer courses that fill the database about Paderborn with high quality content by students.

This leads us to the application that should be prototypical developed within this master thesis, which will be described in the next section.

1.2 WHAT WOULD THE SYSTEM LOOK LIKE (BRIEFLY)

1.3 WHAT WOULD BE BETTER IF THE APP WOULD EXIT? WHO WOULD BENEFIT?

1.4 OUTLINE

The second chapter will explain all needed fundamentals of this Master thesis in detail and will show the used frameworks and tools.

The third chapter will outline the application design and describe some general design decisions.

The fourth chapter will show important parts of the actual implementation, used tools and the final **UML!** (UML!) diagrams of the application.

The fifth chapter will show unit tests and the results of a survey that was used to evaluate the meaningfulness of the created diagrams.

The sixth and last chapter will deal with arisen problems and will discuss the development for future work.

2

TECHNICAL AND METHODOLOGICAL BACKGROUND

After we have seen the need for the planned application, we will now get an insight into the used technical frameworks respectively tools and methodological concepts, which are used during the development process.

2.1 AGILE DEVELOPMENT – SCRUM

The main idea of agile development is described within the agile manifesto[1]. Within agile development, the focus is set to frequent software delivery and close customer relationship. Because Scrum is an agile development method, we find similar ideas in the Scrum development process.

The HiP-application will be developed in a Scrum-like fashion to achieve a high efficiency in the small timeframe. However, a full Scrum approach is not possible because the development team is to small. Nonetheless, we will include the ideas and concepts of the Scrum process but will, for example, combine different roles in one person. But to get a first intuition about the Scrum process itself, we will now describe the main ideas of Scrum and agile development in general.

Using Scrum means, the application will be developed within autonomous short *sprints* with a length between 1 up to 4 weeks. After every sprint the product should be more refined (Schwaber and Beedle (2002)). However, it should be possible to execute the application at the end of any given sprint, which will result in a fast and stable development process. The general development process is also shown in Figure 1. The Product-Backlog is the foundation of every sprint-backlog, which includes every feature that should be added in a specific sprint. In addition to that, daily Scrum meetings should ensure that the whole team is up-to-date and as efficient as possible. In more detail, Scrum is known

1 The whole manifesto can be found here: http://agilemanifesto.org

to reduce every category of work (i.e., defects, rework, total work required, and process overhead) within a **CMMI!** (CMMI!) compliant development process by almost 50% (Sutherland (2009)), which is also great for development in this short timeframe.

The development process also influences the order of the development because the chose of Scrum indicates that we will develop the front-end and back-end in parallel. We will use a **TDD!** (TDD!)-like approach within every sprint (where possible). So, we will at first create needed test cases and afterwards implement against these test cases. This approach will prevent that testing of the application will be shifted into the last week(s) of the development process and done in a superficial way. In addition to that, a comprehensive test suite is a great basis for further development (Maximilien (2003)).

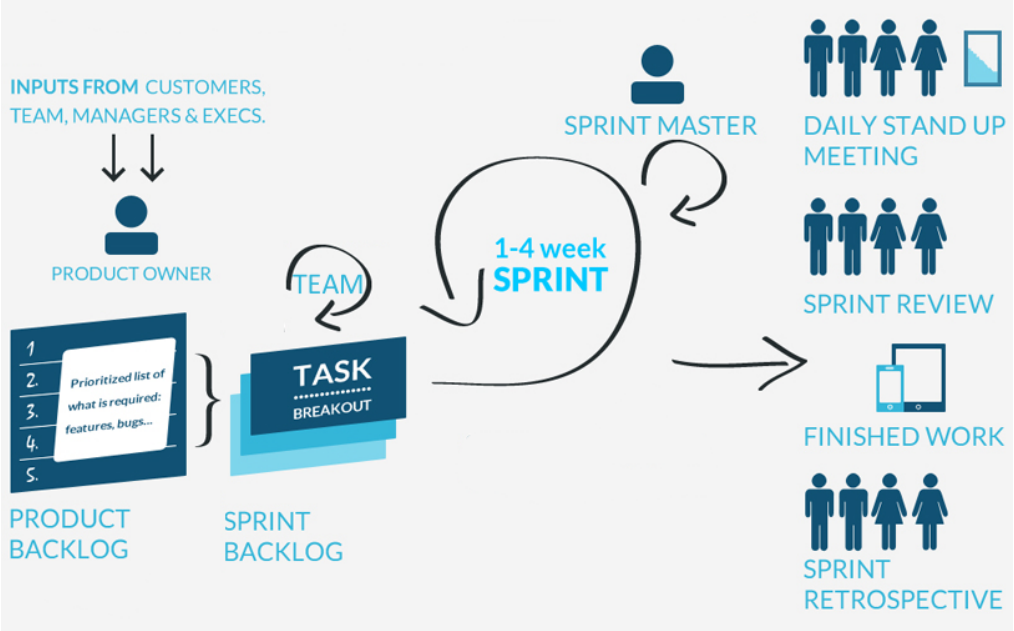

Figure 1: The diagram shows the Scrum development process. (Simplified, original work from **?**)

Now, after we have seen the general concept of Scrum, we will now take a look at methods for cost estimation.

2.2 METHODS FOR COST ESTIMATION

To get a feasible estimation of the workload of a given backlog, as it has been described in section 2.1, we need some methods to create a good work- respectively cost-estimation.

2.3 USED FRAMEWORKS

Because the time frame for the project is quite small, it is not possible to create the whole application from scratch and, thus, we need a couple of frameworks to accelerate the process. We will use the Play-Framework in the backend, which offers a **REST!** (REST!)-interface and handles the routing from **HTTP!** (HTTP!)-requests to application code. On the frontend-side, AngularJS will be used to create a fast and responsive web-interface and Junaio-will be used to include the AR-functionality on the smartphone application.

2.3.1 Play Framework - Backend

We will use the Play framework for the backend of the application because Play is an open source web application framework, which is written in Scala and Java, follows the **MVC!** (MVC!) architectural pattern and handles the routing from HTTP!-requests to application code.

A simple example for the routing configuration file is shown in Listing 2.1. In this file, each documented route consists of an **HTTP!** method and **URI!** (URI!) pattern that is linked to a call of a, so called, *action method* within the Java respectively Scala code. As one can easily see in line 9, it is quite easy to pass parameters to the application code. Furthermore, one can see that the parameters are type-safe; thus, for example, a String passed in as an Integer would result in a compilation error.

Listing 2.1: Simple routing configuration file within the Play Framework

```
1 # Routes
2 # This file defines all application routes (Higher priority
    routes first)
3 # ~~~~
4
5 # Home page
6 GET /                controllers.Application.index()
```

```
7
8  # Usage of parameter
9  GET /thesis/:grade   controllers.Application.exp(grade: Integer)
10
11 # Map static resources from /public to the /assets URL path
12 GET /assets/*file    controllers.Assets.at(path="/public", file)
```

Very briefly, the Play framework includes three different parts:

1) Java Code that implements the controllers. The controllers are used to handle requests that get routed to them via **HTTP!**. A simple controller is shown in Listing 2.2. As one can see within line 10 of Listing 2.2 the String "Your new application is ready" gets passed to the render function of the class index and returned as a parameter within the *ok* function, which creates a simple **HTTP!** header with return-code 200. The index class is a Scala class that gets automatically creates from the Scala/**HTML!** (**HTML!**) template called *index.scala.html*. We will see this in more detail within point 3 of this list.

Listing 2.2: Simple Java-controller within the Play Framework

```
1  package controllers;
2
3  import play.*;
4  import play.mvc.*;
5  import views.html.*;
6
7  public class Application extends Controller {
8
9      public static Result index() {
10         return ok(index.render("Your new application is ready."
               ));
11     }
12 }
```

2) Java Code that implements the model entities. The model is used to do the actual calculation and data handling. In essence, we should include as less application-logic as possible within the controllers and use these model classes instead.

3) Scala templates that are used as *views*. As a return value of the controllers, they pass data to a fitting template and return a corresponding **HTML!**-view. However, we may also skip the template engine sometimes to directly return **JSON!** (**JSON!**)-documents, which can be used to provide a **API!** (**API!**). Listing 2.3 shows the used *index.scala.html* template used in point 1 of this list. In line 1, we declare the used parameters, in our case one String variable, which

we have used to pass the String "Your new application is ready". The *@main* command in line 3 calls another template, which includes everything beside the **HTML!** body. The body of the file is now included in line 4 by calling another framework specific method, which includes a welcome and documentation message and renders our passed String variable.

Listing 2.3: Simple Scala template within the Play Framework

```
1 @(message: String)
2
3 @main("Welcome to Play") {
4     @play20.welcome(message, style = "Java")
5 }
```

As another important fact, Play emphasizes the usage of the **REST!** principle, as it can be seen within the routing configuration file. We can easily and directly make use of the different **HTTP!** commands and use them to structure our **API!** accordingly. In general, **REST!** (**REST!**) is a style of software architecture that is used to build distributed systems and has been introduced in the dissertation of Fielding (2000). As Rodriguez et. al. point out, **REST!** based web services are easier to use than **SOAP!** (**SOAP!**) and **WSDL!** (**WSDL!**)-based ones and getting more and more importance since mainstream web 2.0 service providers are taking up on **REST!**(Rodriguez (2008)). Furthermore, the Play-framework comes with integrated unit testing and full support of asynchronous I/O. So, all in all, Play will noticeably enhance the development speed of the backend.

2.3.2 AngularJS - Frontend

After we have now seen the basics of the Play framework, which will handle the backend functionality, we will now take a look at *AngularJS*, which will provide needed features to create a fast and responsive frontend.

shows expressions

show ng-class

2.3.3 Twitter Bootstrap

Twitter Bootstrap[2] is an open and freely available collection of tools on the basis of the **HTML!**, **CSS!** (**CSS!**) and **JS!** (**JS!**) and can be used to support and accelerate the building of web applications. We will use Twitter Bootstrap inside the user front-end of our web application because it works nicely together with AngularJS and can for example be used to create tabs and alerts. Furthermore, Twitter Bootstrap is nowadays used a lot by common web-applications and, thus, we enhance the external consistency of the **HiP!** (**HiP!**)-application in respect of other web-applications, which may be well known to the user.

ref external consistency

Twitter Bootstrap is licensed under the terms of the Apache License v2.0[3].

2.3.4 Junaio - Frontend

The AR-functionality will be offered by the framework Junaio. The company Metaio, which runs Junaio, offers a developer program to develop own applications on the basis of the Junaio (eco-)system. Moreover, it is completely free of charge for the developers (Junaio (2014). However, deployed apps will be shipped with a Metaio watermark inside as long as you do not buy a specific license.

2.4 WEBGL

Furthermore, we will need WebGL to create a possibility to render and manipulate the 3D-point clouds, of the scanned objects, right within the browser.

2.5 TOOLING

A couple of frameworks and techniques is a good start for creating such a sophisticated system, however, we will also need fitting tooling to support the development. These tools will be described in the following section.

2 Twitter Bootstrap is hosted on GitHub and can be downloaded here: https://github.com/twitter/bootstrap

3 The terms of the license can be found here: http://www.apache.org/licenses/LICENSE-2.0

2.5.1 Git

We will use Git, which is a commonly used distributed revision control and source code management system, for the versioning of our source-code. Git is free software distributed under the terms of the GNU General Public License version 2.

The service GitHub offers his users the possibility to maintain public and private Git repositories. The usage of GitHub is free, if the user uses public repositories only. We will use GitHub to host our source-code.

2.5.2 Jira

2.5.3 IntelliJ IDEA

IntelliJ IDEA is a Java **IDE!** (**IDE!**) by the company JetBrains.

The current version offers support for Java 8, UI designer for Android development, Play 2.0 and Scala and is, thus, a good choice to work with because all of these features will be used in our development process.

The **IDE!** is available as an Apache 2 Licensed community edition and a commercial edition.

2.6 TESTING TECHNIQUES

2.6.1 Why is testing within an agile development even more important

2.6.2 TDD

2.6.3 The fundamental test process

3

DRAFT OF THE APPLICATION

To handle all these problems that have been described above within the section about the current situation, we will create a smartphone application that gets supported by a backend web-server. The master thesis will handle the planning respectively cost estimation of the different parts/features of the system and will, after the needed technologies/frameworks are elaborated and evaluated, include a prototypical implementation of the needed components. But at first, we will start with the first step in a development process; the requirements engineering.

3.1 REQUIREMENTS ENGINEERING

Because the HiP-Application will be developed closely together with our *costumer*, other working-groups at the university of paderborn, the whole process starts with the requirements engineering phase.

First of all, a requirement is defined as "[...]A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents[...]" (IEEE (1990)).

We started the development with a requirements engineering meeting together with our *customer* and ended up with a couple of cards with written user stories. Afterwards, these stories got refined to concrete requirements, which are measurable and prioritized. A complete list of all requirements, which were derived from these user stories, can be found within the appendix in Table 5 and **??**. These requirements can directly be used to derive test cases from them, which is good because we will use a **TDD!** driven development approach in every sprint.

In the following, we will explain the two parts of the application separately.

3.2 BACKEND (WEB-SERVER)

Another important part of the system is contained within the backend-web-server. The backend should contain the whole data handling and assessment. The students should be able to add data to the system (e.g., a textual article, graphics, AR-data, etc.) and to modify existing data via a **CMS!** (**CMS!**). These entries get reviewed, for example by the course supervisor, and unlocked for the frontend application. To do this, the backend needs features like annotations and highlighting, which should be private for a specific user. By using this, the supervisor can evaluate the given texts right within the **CMS!** and give his final judgement. If the supervisor is not satisfied with the quality of the given text, he should be able to send the document back to the student, to get a revised and updated version of the document. If the supervisor is satisfied, he can unlock the information for showing in the frontend application.

The data should be stored in a way that it can be shown within an AR-environment in the smartphone application. Of course, we will need some mechanism to structure the data, for example tags or stored categories. This kind of information (especially tags) are also very important for the described filtering techniques on the client side.

Furthermore, the backend should include a way to modify the point-clouds of the objects that has been scanned with the smartphone application. It will need features to add annotations directly to these point-clouds to show them afterwards within the AR-environment. This editor will be created on the basis of **HTML5!** (**HTML5!**) and **WebGL!** (**WebGL!**). A mockup of this site is shown in Figure 2. These annotations should also be assessable and (un-)lockable for the supervisor.

Figure 2: A mockup showing the augmentation editor that will be included in the web-application. The editor will be used to edit the point-clouds, which have been added with the help of the smartphone-application

3.2.1 Cost estimation of the backend

3.2.2 Input data/content via CMS in the system

3.2.3 Manage content as a reviewer

3.2.4 Including a 3D-Tooling system for point-clouds (WebGL)

3.3 FRONTEND (APP)

The smartphone application is the part of the system that gets shipped to the end-user (respectively downloaded via an App-Store like Google-Play). The user can use the app to find interesting places respectively objects in Paderborn and is able to start a navigation to the place/object easily. Furthermore, the user can get an overview about all places in Paderborn by activating a map

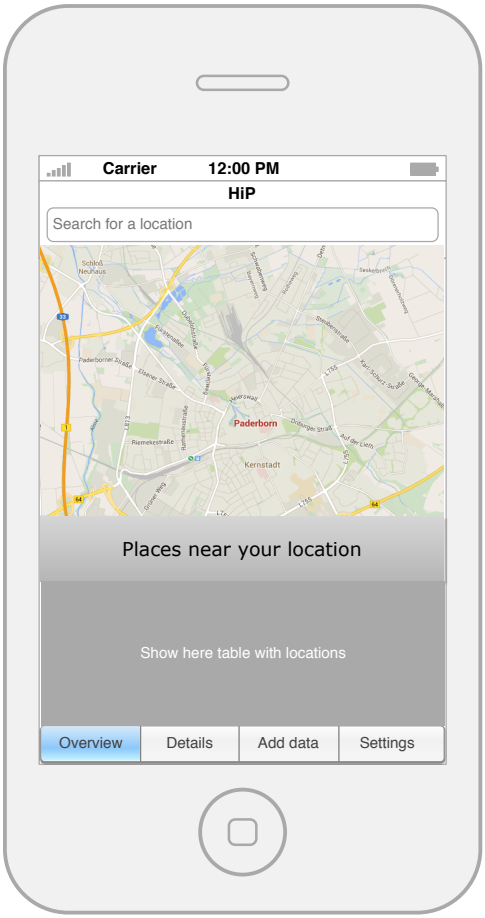

Figure 3: A mockup showing the main page of the frontend application showing a map of paderborn and a general overview about the UI-elements

that shows all entries within the system. A mockup of this view is shown in Figure 3. Of course, the user will be able to set up specific filters like 'show only art', 'show only historic buildings' or 'use simplified language' to adapt the system to his own experiences and educational qualifications. Moreover, if the university courses would add information over years, the system will need filtering features like this to handle the complexity of the data.

After an user has reached an interesting place, he can use the details tab to switch into the AR-mode. With this view, the user can use the smartphone-camera to embed information, which has been added via the backend, right into the picture of the object. An mockup of this view is shown in Figure 4. To create a feasible input for the AR system, the user should be able to scan objects in 3D right with his smartphone application and send the data (i.e., a

point-cloud of the scanned object), back to the web-server. Afterwards, the user can add annotations to the point-cloud via the web-backend of the system.

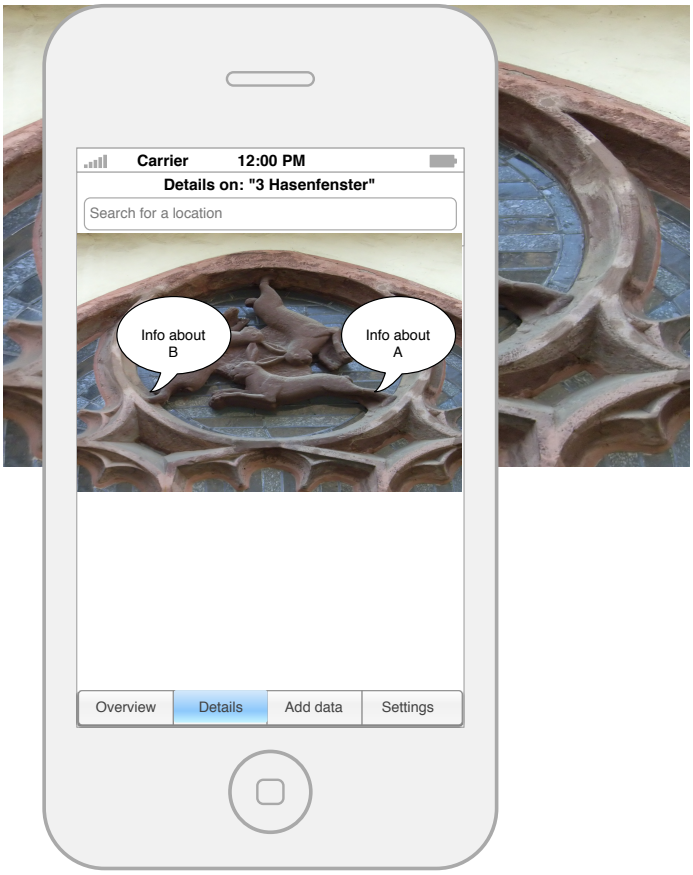

Figure 4: A mockup showing the details page of the "Dreihasenfenster" while the camera of the smartphone is pointing to the window itself

3.3.1 Cost estimation of the frontend

3.3.2 Input data into the system (scan objects and annotate them)

3.3.3 Show close "interesting places" within a map / via a overlay

3.3.4 Navigation to "interesting places"

3.4 INTERFACE

3.4.1 Data format for AR files

4

IMPLEMENTATION DETAILS

5

TESTING THE APPLICATION

5.1 TEST ENVIROMENT

5.2 TESTING RESULTS

5.3 ACCEPTANCE TEST OF THE PROTOTYPE

5.3.1 Small usablity study of the app

5.3.2 Small usability study of the backend / CMS

6

DISCUSSION AND FUTURE WORK

6.1 ARISEN PROBLEMS WITHIN THIS THESIS

6.2 DISCUSSION AND FUTURE WORK

6.2.1 Results / Conclusion

6.2.2 Future work

A

APPENDIX

The appendix contains some diagrams and tables that were to big to put them into the continuous text.

ID	Description	Acceptance criteria
BS1	The supervisor should draft guidelines and assistance (e.g., Button with question-mark)	- At least one information resp. help function per functionality
BS2	The supervisor should be able to see which data is missing	
BS3	The supervisor should be able to see data that is ready for review	- Try without content that is ready for re - Try with content that is ready for revie
BS4	The supervisor should be able to assign exhibits to students	- Try assigning an exhibit to one student - Try assigning an exhibit to more than s
BS5	The supervisor should be able to trace content back to specific students	
BS6	The supervisor should be able to define topics and exhibits	
BS7	The supervisor should be able to comment and discuss the given content of the students	
BS8	The supervisor should be able to mark errors in the content	- Try marking an error twice
BS9	The supervisor should get e-mail notifications about new content handed in by students	- The e-mail should be received in less than 2 minutes in 90% of the time
BS10	The supervisor should be able to copy topics and categories (e.g., usage of templates for different typical cases, duplication, etc.)	
BS11	The supervisor should be able to define validation-constraints (e.g., character limitation)	- Try with error within the validation constraints
BSt12	The supervisor is able to see the amount of texts and pictures in a hidden topic	- Try without any content included - Try with a lot of content included
BS13	The supervisor should be able to work offline	

Table 1: Showing the derived requirements of the Backend for the supervisor role, which are sorted by priority

ID	Description	Acceptance criteria	Pri
BSt1	The students are only able to send in specific content (field / topic)	- Try sending content to another topic	1
BSt2	The students should get an e-mail notifiation about new content in their topic (e.g., send in via fellow students)	- The e-mail should be received in less than 2 minutes in 90% of the time	1
BSt3	The students should be able to send in metadata	- Try with errors within the meta-data	1
BSt4	The students should be able to overview the possible links within their topic (e.g., GPS-information)	- Try without any links - Try with a lot of links	1
BSt5	The students should be able to send in content		1
BSt6	The students should be able to propose topics and content		1
BSt7	The students should only have temporary access to the backend	- Try logging in after the temporary account has been deleted	1
BSt8	The students should have access to all temporary content (i.e., not reviewed content)		1
BSt9	The students should be able to create interdisciplinary groups and communicate within these		1
BSt10	The students should be able to see their content in a preview mode that simulates the frontend		2
BSt11	The students should be able to see content of other groups in a preview mode that simulates the frontend		2
BSt12	The students should be able to comment and discuss the content of their group or other groups		2
BSt13	The students should be able to hide their unfinished work to the supervisor	- Try hiding without having any content	2

Table 2: Showing the derived requirements of the Backend for the student role, which are sorted by priority

ID	Description	Acceptance criteria	Priority
BM1	The master should be able to recover data by using a back-up system	- The recovery should not take longer than one hour	1
BM2	The master role can be assigned to a couple of users at the same time		2
BM3	The master is able to do the final acceptance		2

Table 3: Showing the derived requirements of the Backend for the master role, which are sorted by priority

ID	Description	Acceptance criteria	Priority
BMi1	The data of the system is stored on IMT-Server		1
BMi2	The system can be updated and maintained in the future (e.g., project-groups, SHK, etc.)		1
BMi3	The content should not be limited to specific layouts, views (e.g., languages) and templates		1
BMi4	The system should be expandable (e.g., new content, filters, etc.)		1
BMi5	The system should be safe with respect to hackers resp. data manipulation	-The system should be safe with respect to the economic view/ definition of safety	1
BMi6	The system offers features to manage groups		2

Table 4: Showing the derived requirements of the Backend, which are sorted by priority

ID	Description	Acceptance criteria	I
F1	The user should be able to navigate to the different locations shown in the HiP-application		1
F1.A	The user should be able to navigate to the different locations and discover these locations on his own		1
F1.B	The user should be able to navigate to the different locations and use round tour information of the application		1
F1.B	The user should be able to navigate to the different locations while using filters (e.g., epochs)		1
F2	The user should be able to create thematic routes	- Try creating a route without assigning a theme	1
F3	The user should get a list of locations/exhibits in Paderborn		1
F4	The user should see linkings within an exhibit different exhibits (e.g., Liborischrein -> Hle -> Scriptorium)		1
F5	The user should be able to deselect specific categories	- Try deselect only one - Try deselect many	1
F6	The user should be able to filter exhibits on the map (e.g., locations, historical figures, etc.)		1
F7	The user is able to overlay the current map of the city with historical maps	- Try overlay one map with a hist. one - Try overlay a couple of maps	1
F8	The user is able to see himself and historical places on the map		1
F9	The user should not exceed his storage on the smartphone		1
F10	The user should not exceed his data-volume on the smartphone		1
F11	The user should be able to use the application easily (good usability)		1

Table 5: Showing the derived requirements of the Frontend, which are sorted by priority

ID	Description	Acceptance criteria	Priority
F12	The user should be able to switch between different contents (e.g., Video, 3D, etc.) fast		1
F13	The user should be able to see *invisible* objects within the details-tab (e.g., something placed inside an altar)		1
F14	The user should be able to use tablets and smartphones		1
F15	The user should only get details about an exhibit while he is next to it or afterwards	- Try to get details beforehand	1
F16	The user should be able to get texts, graphics/ pictures and links about an exhibit		1
F17	The user should be able to get audio, video and 3D-views/models about an exhibit		2
F18	The user can create and join treasure hunts respectively geo-caching features		2
F19	The user should get informed about exhibits and locations that are next to him		2
F20	The user should be able to get navigated with AR-rabbits		2
F21	The user should be able to get navigated inside of a building		2
F22	The user should be able to choose between different starting possibilities (i.e., tour, discovery and historical topics)		2
F23	The user should be able to hear the content via an audio-guide		2
F24	The user should be able to get exhibits as comparison by using AR		2
F25	The user should be able to create own notes and comments		2
F26	The user should be able to share content via social media		2

Table 6: Showing the derived requirements of the Frontend, which are sorted by priority

ID	Description	Acceptance criteria
F27	The user should be able to export content as PDF and create book-marks	- The export should not take longer than 30 se in 90% of the time
F28	The user should be able to get the content in different languages (i.e., englisch, french, turkish)	
F29	The user should be able to choose between different criteria with respect to the audience (e.g., different ages of people)	- Try selecting one criterion - Try selecting more than one criterion

Table 7: Showing the derived requirements of the Frontend, which are sorted by priority

BIBLIOGRAPHY

Azuma, R. T. (1997). A survey of augmented reality a survey of augmented reality a survey of augmented reality. *In Presence: Teleoperators and Virtual Environments*, 6(4):355–385.

Bondo, J.Barnard, D. B. D. (2010). *iPhone User Interface Design Projects*. Apress.

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, USA. AAI9980887.

Hampel, T. (2001). *Virtuelle Wissensräume*. PhD thesis, Universität Paderborn.

IEEE (1990). Ieee standard glossary of software engineering terminology. *IEEE Std 610.12-1990*, pages 1–84.

Junaio (2014). Why to become a junaio developer. Slideshare presentation, http://www.slideshare.net/metaio_AR/why-to-become-a-junaio-developer, last checked 14.09.2014.

Maximilien, E. Michael. Williams, L. (2003). Assessing test-driven development at ibm. In *Software Engineering, 2003. Proceedings. 25th International Conference on*, pages 564–569. IEEE.

Rodriguez, A. (2008). Restful web services: The basics. *Online article in IBM DeveloperWorks Technical Library*, 36.

Schwaber, K. and Beedle, M. (2002). *Agile Software Development with Scrum*. Pearson.

Statista (2014). Global apple iphone sales from 3rd quarter 2007 to 3rd quarter 2014 (in million units). Website, http://www.statista.com/statistics/263401/global-apple-iphone-sales-since-3rd-quarter-2007/, last checked 13.09.2014.

Sutherland, J. Jakobsen, C. (2009). Scrum and cmmi – going from good to great. *Agile Conference*.

Wikimedia-Foundation (2014). About us. Website, https://www.wikimedia.de/wiki/%C3%9Cber_uns, last checked 12.09.2014.

Wikitude (2014). Augmented reality for multiple platforms. Website, http://www.wikitude.com/products/wikitude-sdk/, last checked 14.09.2014.

STATUTORY DECLARATION

I hereby declare that I have developed and written this thesis on my own, and no external sources were used except as acknowledged in the text and footnotes. The thesis in this form or in any other form has not been submitted to an examination body and has not been published.

Paderborn, October 16, 2014

Jörg Amelunxen