

[Logo] Game

# Table of Contents

1. Introduction and Goals .....	1
2. Requirements and Overview .....	2
3. Presentation of the project team .....	2
3.1. Eric Haneder .....	2
3.2. Jan Binder .....	2
4. Division of Tasks .....	3
5. Qualitygoals .....	4
6. Stakeholders .....	5
7. Theoretical task: Eric Haneder (Learning) .....	6
7.1. Learning .....	6
7.2. Physiological structures of learning .....	6
7.3. Learning type classification according to Vester .....	6
7.4. Learning preferences according to Dunn .....	8
7.5. Learning styles according to Pask .....	9
7.6. Learning styles according to Kolb .....	9
7.7. Self-regulated learning .....	11
7.8. Gamificated Learning .....	11
7.9. Conclusion .....	12
8. Theoretical task: Jan Binder (Knowledge management) .....	13
8.1. Knowledge .....	13
8.2. Knowledge management in context of the knowlegde socitey .....	13
8.3. Knowledge management models .....	13
8.4. Knowledge management methods and tools .....	13
9. System, Scope and Context .....	22
9.1. Technical Context .....	22
9.2. Description .....	23
9.3. Considerations .....	23
10. Solution Strategy .....	24
10.1. Building from Scratch .....	24
10.2. Programming Language .....	24
11. Design Decisions .....	25
11.1. ADR001-BuildingFromScratch .....	25
11.2. ADR002-Technology stack .....	27
11.3. ADR003-Build and dependency managment .....	28
11.4. ADR004-UI technology stack .....	29
11.5. ADR005-Communication with distributed systems .....	30
11.6. ADR006-Database .....	31
11.7. ADR007-Java EE application server .....	32

11.8. ADR008-Standard code formatter .....	33
12. Practical task: Eric Haneder (Front end) .....	34
12.1. Beginning .....	34
12.2. Fundamentals .....	34
12.3. Getting Started .....	37
12.4. User-Interfaces .....	38
12.5. Java classes .....	47
13. Practical task: Jan Binder (Back end) .....	51
13.1. Beginning .....	51
13.2. Fundamentals .....	51
13.3. Getting Started .....	52
14. Bibliography .....	64
15. List of figures .....	64
16. Timetables .....	66
16.1. Eric Haneder .....	66
16.2. Jan Binder .....	70

---

# 1. Introduction and Goals

The ARZ wants to establish a new way of gamified education for the members. ARZ means "Allgemeines Rechenzentrum" and it is an IT service provider with locations in Innsbruck and Vienna. Their homepage: <https://www.arz.at/>

The workers often have different knowledge about programming, so it is rather time consuming and costly to further educate them.

This is where the idea of "GAME" comes into play. The "GAME" is intended to be a platform on which employees can improve their programming knowledge with little games, quizzes or competitions.

A great focus lies on the terms "self-study" and "Gamification". Gamification means that the information is transmitted through games to ease the learning process.

Furthermore there should be a possibility for trainers to bring in their own ideas, quizzes or modules. One of the goals of this diploma thesis is to motivate the people to continue their education in their free time. There should be a possibility to learn some skills without much effort.

Another important aspect of "GAME" is the scoring system, which will be implemented in the software. This will create transparency for employees and employers. This scoring system will be a great motivation for workers.

## *The most relevant Stakeholders*

- Roland Moder
- Konrad Renner
- Michaela Würzl
- Cornelia Sammer



The expectations of the Stakeholders should be fulfilled.

The expectations of the stakeholders regarding the thesis are:

- sufficient documentation regarding the arc42 template
- Fulfillment of the quality goals
- a clear and well-functioning software
- an easy-to-use user interface

## 2. Requirements and Overview

These are the requirements the system must meet:

- the system must be created/modified to get a simple, easy to use education platform
- trainers should be able to upload their own tasks and games to the platform
- it should contain a variety of difficulty levels
- there must be a point system which rewards the user and lets him compare his points with other users
- players should be able to track their progress

## 3. Presentation of the project team

### 3.1. Eric Haneder

- Place of residence: Untergrub
- Born: 17.05.2001
- Education
  - Volksschule Göllersdorf
  - Hauptschule Göllersdorf
  - HTL Hollabrunn, Wirtschaftsingenieure-Betriebsinformatik

### 3.2. Jan Binder

- Place of residence: Sonnberg
- Born: 09.04.2001
- Education
  - Volksschule Hollabrunn
  - Bundesgymnasium Hollabrunn
  - HTL Hollabrunn, Wirtschaftsingenieure-Betriebsinformatik

## 4. Division of Tasks

Eric Haneder is in charge of the functions and design of the website. He is responsible for ensuring that the homepage is clear and easy to use. He programs the website on the basis of *Java Enterprise Edition*, *\_PrimeFaces* and *HTML*, as requested by the company.

Jan Binder is responsible for programming the backend. This includes that the connection to the database is established, the database contains the correct data and the way the website can communicate with the database. He programs with the help of *JPA* and *SQL*.

## 5. Qualitygoals

The four quality goals for the architecture whose fulfillment is of highest importance to the major stakeholders

Table 1. Quality Goals

Name	Description
System ready and usability	The goal of this project is to create or modify a system to get a learning platform for the ARZ members. So the most important quality goal is the completion of the system because there cannot be any other features if the basic system does not exist. Also the system should be easy to use/understand and should not have flaws in it.
The point system	One of the most important goals of the education platform is to implement a point system for the users. The point system rewards the ARZ members and the users should be able to compare their points with each other. This will boost the motivation of the learners to educate themselves even more, by for example trying to reach certain points.
Trainers can upload games and tasks	Another feature should be the possibility for selected trainers to upload their own tasks and games for the users of the education platform. This should be the foundation to never run out of new games and tasks.
Up to date documentation is released with every update of the system	The documentation of the whole project is another goal which is very important for the successful completion. It should be without grammar or writing errors and should have an overall good looking format.

## 6. Stakeholders

The stakeholder are the following:

- **Thesis supervisor** (contractor side): Roland Moder [roland.moder@aon.at](mailto:roland.moder@aon.at)
- **Thesis supervisor** (client side): Konrad Renner [konrad.renner@arz.at](mailto:konrad.renner@arz.at)
- **Human resource development:** Michaela Würzl [michaela.wuerzl@arz.at](mailto:michaela.wuerzl@arz.at), Cornelia Sammer [cornelia.sammer@arz.at](mailto:cornelia.sammer@arz.at)

### *Expectations:*

#### **Roland Moder**

He expects the diploma thesis to run smoothly and be completed within a reasonable time, this means it has to be completed by the end of April. Additionally, the diploma students should fulfill the requirements for a diploma thesis. Furthermore, there should be a sufficient documentation of the work done.

#### **Konrad Renner**

He expects the software to contain a scoring system that records the learning progress of every member, which can be evaluated and compared. Furthermore, he expects to be able to maintain the tasks independently, this means if all tasks have been fulfilled on the platform, there should be an opportunity for coaches to be able to put in new tasks.

#### **Michaela Würzl and Cornelia Sammer**

They expect a timely completion of the project, so that the concept is completed by 31.12.2019, and the implementation should be completed by the end of April. Moreover, the software should be utilizable and integradable, this means the user interface should be easy to use for employees, trainers and moderators.



# 7. Theoretical task: Eric Haneder (Learning)

## 7.1. Learning

Everyone has to learn, both actively and passively. Whether at school or in professional life, further education is part of the essence of man. The term "learning" usually means the acquisition of knowledge, the development of skills or the practice of motor processes. However, "learning" also refers to activities that are carried out with the aim of changing internal conditions.

*(vgl. Selbstreguliertes Lernen in der dualen Ausbildung - Lerntypen und Bedingungen, Johannes Rosendahl, 2010)*

## 7.2. Physiological structures of learning

Regardless of the theoretical approach to learning, it is true that learning is linked to the ability to modify the central nervous system (CNS) (Birbaumer & Schmidt 1996, 2006). The central nervous system consists of the brain and spinal cord, whose impulse-transmitting cells (neurons) receive and process signals from the peripheral nervous system and generate and transmit signals for the regulation of organs such as muscles.

Despite the large amount of signals that are constantly received by the sense organs from the environment and transmitted to the CNS, the number of connections between the peripheral nervous system and the CNS is relatively small. Most of the estimated 100 billion neurons (Goldstein 1997, S. 11) serve to process information (pattern recognition, linking), they receive their input from cortical neurons and deliver their output to other cortical neurons.

Learning is based on the plasticity of synapses and dendrites. Plasticity includes activation or breaking of connections between nerve cells, elimination of blocking connections, changes in synaptic density, and branching and thickening of dendrites. Not one but several of these processes are involved in the various forms of learning.

*Dendrites:* root-like branches that make up the impulse-transmitting cells of the CNS

*(vgl. Selbstreguliertes Lernen in der dualen Ausbildung - Lerntypen und Bedingungen, Johannes Rosendahl, 2010, S.22)*

There are many people who have tried to classify different learning preferences and learning types. I will discuss these classifications in more detail below.

## 7.3. Learning type classification according to Vester

In the book "Denken, Lernen, Vergessen" (Vester, 1998), Frederick Vester defined four types of learning. However, his book was interpreted differently, so that different versions of his classification can be found. At least on the fact that there is an auditory, a visual and a haptic learning type, there is agreement. Though there are some people who think that the fourth learning type after Frederik Vester is the communicative learning type and others think that it is an intellectual learning type.



One also comes across terms like "cognitive learning type" "intellectual learning type", or the "kinesthetic learning type" "haptic learning type."

*Evidence for a classification according to Vester with a "communicative learning type"*

<https://www.teko.ch/die-vier-lerntypen-und-ihre-besonderheiten;>  
<https://www.mystipendium.de/studium/lerntypen;>  
<https://karrierebibel.de/lerntypentest/#Die-vier-Lerntypen-angelehnt-an-Frederic-Vester;>  
[https://learnsolution.de/die-vier-lerntypen-nach-frederic-vester/;](https://learnsolution.de/die-vier-lerntypen-nach-frederic-vester/)

*Evidence for a classification to Vester with an "intellectual learning type"*

[http://www.rechtschreibwerkstatt-konzept.de/wp-content/uploads/2015/02/Looss\\_Lerntypen.pdf;](http://www.rechtschreibwerkstatt-konzept.de/wp-content/uploads/2015/02/Looss_Lerntypen.pdf)  
[https://wb-web.de/wissen/lehren-lernen/lernstile-und-lerntypen.html;](https://wb-web.de/wissen/lehren-lernen/lernstile-und-lerntypen.html)  
[https://smarter-learning.de/lerntypen/klassische-lerntypen-nach-vester/;](https://smarter-learning.de/lerntypen/klassische-lerntypen-nach-vester/)  
[https://wissenschafts-thurm.de/lerntypen/;](https://wissenschafts-thurm.de/lerntypen/)  
"Erforschung der Lerntypen und -strategien am Beispiel einer Handelsschulklasse",  
Petra Hochleitner, 2016

I will be analyzing this classification according to Vester:

1. optical/visual → Learning through seeing and observing
2. auditory/acoustically → Learning through listening and speaking
3. haptisch → Learning through action (touching and feeling)
4. cognitive → Learning through recognition/intellect

### **7.3.1. Auditory learning type**

The auditory learning type learns while using his hearing. He/She has no problem listening to someone over long periods of time. The auditive type prefers auditioned learning content and learns better if he/she reads the text aloud himself/herself.

### **7.3.2. Visual learning type**

This learning type learns the best, when he takes up the learning content through his eyes. Reading texts will lead to great learning achievements. This gets a better understanding of facts through looking at pictures.

### **7.3.3. Haptic learning type**

The haptic learning type achieves the best learning succes if he cant feel the information he has to learn. "Learning by Doing" describes this lerning method pretty good. He prefers to be actively integrated into the learning process. Practical demonstrations are helpful too.

### 7.3.4. Cognitive/Intellectual learning type

This learning type understands and saves information the best, by thinking about and critically examining the information. The perception channel not important for taking up learning content.

(vgl. "<https://smarter-learning.de/lerntypen/klassische-lerntypen-nach-vester/>"; letzter Zugriff: 14.01.2020)

### 7.3.5. Connections

Most of the time, it is not possible to assign someone a certain learning type, as there mix-types of learning. There are some people who learn faster by reading through text and writing it down afterwards (visual & haptic), and others are better of learning by reading and reciting out loud (auditory & haptic).

So if you want to learn effectively, you should try to appeal to more than one sense, to find a combination that fits your style.

#### How are different types of learning formed?

It would be a lot easier, if everybody could learn the same way. In reality, there are many people with different learning types, this is due to changes in personal characteristics, habits and previous experiences.

#### So how can you find out which learning type suits you best?

On the Internet you can find many so-called "learning type tests". In these tests, you have to answer a few simple questions, that depict typical life situations. Here are two hyperlinks, which lead to easy tests: \* <http://arbeitsblaetter.stangl-taller.at/TEST/HALB/Test.shtml> \* <http://www.philognosie.net/index.php/tests/testsvew/150/>

After completing one of these tests, you will receive a recommendation as to which type of learner is more suitable for you.



It should be noted that different tests can also lead to different recommendations. This is due to the fact that the tests have different focuses and are more or less comprehensive.

If you do not want to take such tests, you can just look up different well-approved learning methods and try them yourself. Check how good you can remember something while using all of the different learning methods either alone or combined.

(vgl. "Erforschung der Lerntypen und -strategien am Beispiel einer Handelsschulklasse", Petra Hochleitner, 2016)

## 7.4. Learning preferences according to Dunn

Dunn and Price (1989) defined learning style as a typical way of learning that is influenced by different elements of the environment. This regards:

- physical stimuli (light, sound, temperature, design)
- social stimuli (pairs, peers, adults, groups)
- stimuli of learning material (auditiv, visuell, taktil, kinästhetisch)
- emotional stimuli (responsibility, persistence, motivation, disciplin)

These factors are measured by the "Learning Styles Inventory". However, this model takes little account of the actual cognitive processes that play a role in learning.

(vgl. *"Lernorientierungen, Lernstile, Lerntypen und kognitive Stile"*, Ulrike Creß, in *"Handbuch Lernstrategien"* von Heinz Mandl & Helmut Felix Friedrich, S.373)

## 7.5. Learning styles according to Pask

Around 1972, Pask and Scott identified two opposing learning strategies used in problem-solving tasks where people had to search for information independently. They described the consistent usage of these strategies as a learning style.

The holistic strategy means that learners always keep the big picture in mind and only turn to detailed questions in a second step. If this strategy is applied consistently, Pask speaks of the learning style of comprehension learning. On the other hand, learners with a serial strategy work their way step by step through the learning material and primarily turn to individual questions. If this strategy is used consistently, Pask speaks of operation learning. Both strategies can lead to the same success.

In their extreme form, however, both have a negative effect on performance, which is why Pask assigns both learning styles to corresponding learning pathologies. *Globetrotting* refers to the learning pathology of extreme comprehension learning, in which learners make inadmissible generalizations without the corresponding individual analysis. *Improvvidence* describes the extreme form of operation learning, in which people lose themselves in details without being able to connect them to a big picture. Since the differences between holistic and serial approaches affect not only learning behaviour but the entire way in which information is sought and processed, they are often interpreted as cognitive styles.

(vgl. *"Lernorientierungen, Lernstile, Lerntypen und kognitive Stile"*, Ulrike Creß, in *"Handbuch Lernstrategien"* von Heinz Mandl & Helmut Felix Friedrich, S.369)

## 7.6. Learning styles according to Kolb

In 1984 David Kolb took a completely different approach to classifying learning types. According to Kolb, the learning process is based on two orthogonal bipolar dimensions.

The first dimension depicts how people perceive and collect information. Persons can perceive via the senses through practical experience or through abstract comprehension.

The second dimension represents the way information is processed. It ranges from active trying to mental observation. (orthogonal → two straight lines are called orthogonal if they enclose a 90 degree angle) The following figure shows the dimensions:

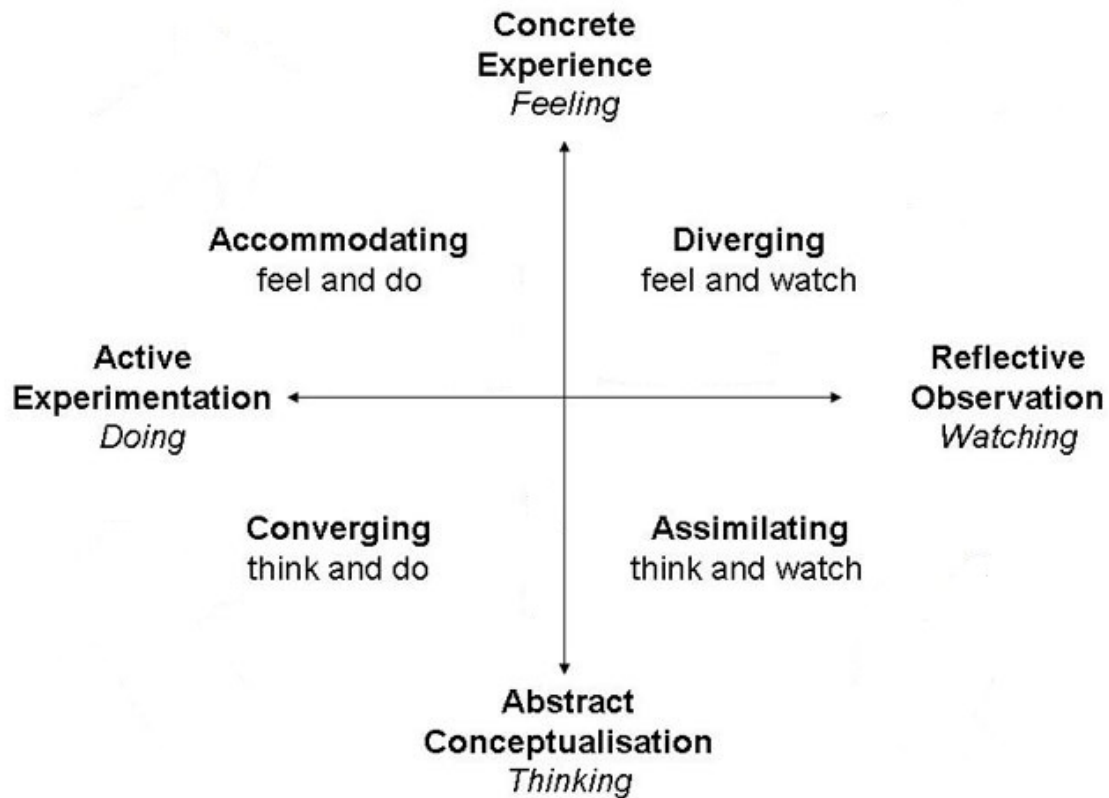


Figure 1. Learning styles according to Kolb

Source: (<https://selfdirectedlearning.webnode.es/learning-styles-by-kolb/>; letzter Zugriff 28.01.2020)

Kolb presents four learning styles defined by the four quadrants that result from these orthogonal dimensions.

*Convergers* explore their environment through active probing and process information in an abstract way. They are therefore interested in testing their theories and solving problems deductively.

*Divergers* combine mental observation with practical experience. This often leads them to creative solutions.

*Assimilators* connect abstract comprehension with mental observation. They are therefore mainly interested in developing abstract theories and defining problems, less in solving concrete problems.

*Accommodators* combine active experimentation with concrete experience. They prefer casual learning directly from the situation. The learning style of a person is measured by Kolbs' Learning Style Inventory (KLSI).

Kolb's approach is by far the most frequently cited of the approaches for recording learning styles.

(vgl. "Lernorientierungen, Lernstile, Lerntypen und kognitive Stile", Ulrike Creß, in "Handbuch Lernstrategien" von Heinz Mandl & Helmut Felix Friedrich, S.371-372)

## 7.7. Self-regulated learning

The concept of self-regulated learning is neither a precisely scientifically defined term nor a uniformly used term in everyday language. Furthermore, the terms self-regulated learning, self-directed learning, learner control can hardly be defined clearly.

Niegemann and Hofer (1997) or Büser (2003) define that in self-directed learning, in contrast to self-directed or self-regulated learning, the learning goal is determined by the person himself. Other authors, on the other hand, see the decision on learning goals explicitly as a component of self-directed or self-regulated learning (Arnold & Gomez-Tutor 2006; Dehnbostel 2003; Lang & Pätzold 2006; Neber 1978; Schreiber 1998, S. 45).

(vgl. *Selbstreguliertes Lernen in der dualen Ausbildung - Lerntypen und Bedingungen*, Johannes Rosendahl, 2010)

In self-regulated learning, the own motivation plays an important role. Here the formation of learning goals can help. In these learning goals personal standards and reference achievements are expressed.

Based on this goal, actions are taken, their execution is monitored and their results are compared with the goal. In response to the comparison result, the actions are either adjusted or stopped. The actual self-regulation includes the interacting sub-processes self-observation, evaluation processes and self-reaction (Bandura 1986, S. 337). During self-observation, the learner registers his performance in terms of various performance dimensions such as quality, originality, quantity and morality as well as correctness and accuracy. Based on this, he will have to take action, as mentioned above. Subsequent results are again observed and evaluated, thus creating a cyclical loop (Bandura 1989).

The term self-regulation emphasizes that a person influences his or her own actions to a considerable extent (Weinert 1982). However, this does not exclude the possibility that the trainer/teacher may carry out an occasional learning check (comparison of target and actual learning). Purely self-regulated action is theoretically possible, such as the creation of an artificial language and its practice through inner speech, but in professional reality there will usually be a greater degree of external control.

## 7.8. Gamified Learning

Gamified learning is a unique way for learners to remember their learning material. The goal is to maximize enjoyment and engagement through capturing the interest of learners and inspiring them to continue learning (Huang 2013). Gamification in particular means implementing game-elements, to a non-game environment.

Some elements of games that can be used to motivate learners and ease learning include:

- Progress mechanics (points, badges, leaderboards, etc.)
- Immediate feedback
- Increasing challenges
- Social connection

- Opportunities for mastery, and leveling up

When a learning platform contains the use of some of these elements, it can be considered "gamified". A gamified system, which is often used in classrooms, is Kahoot. Kahoot is a website, where the teacher can create quizzes, and let the students participate in them via their smartphones. The quiz gets displayed e.g. with a beamer, so every student can see the questions. The principle is easy: Every student gets to answer the same question in the same timeframe, and whoever answered the fastest, gets the most points. If the student answers incorrectly, he/she will not get any points. At the end of the quiz, a leaderboard is displayed with the best scoring players on the top. This system embraces progress mechanics, immediate feedback and social connection.

## 7.9. Conclusion

Learning is an essential part of the human nature. The human race developed the way it did, because humanity kept learning and improving skills. "You never stop learning.", is an important figure of speech. In modern times learning is as important as ever, because the world changes rapidly.

What we created, was another tool for people to learn new things or check their existing knowledge. We did this in a very unique way. In a *gamified* way. Combining elements of games with learning is proven to improve one's learning capabilities. Even though, there are several different learning types, and many more interpretations of such, our system is embracing many of them. For example, the visual learning type reads through the course and remembers the material better that way. The cognitive learning type learns better by taking the quizzes and creating connections in their head.

I tried to include as many of the things I learned by researching my theoretical part of the diploma thesis, into the platform. I think, the members of the ARZ will have an easy learning experience with the website we created.

# 8. Theoretical task: Jan Binder (Knowledge management)

## 8.1. Knowledge

Knowledge is omnipresent wherever we go. Everyone around has more or less knowledge about every kind of topic.

The definition of knowledge is a problem since the early ages. Philosophers like Aristoteles or Platon, who lived ~400 years before christ, worried about what knowledge really is. It is connected to the search of truth. The approach of Platon says that knowledge is acquired by means of deduction, the absolut truth develops through thinking logically. Aristoteles opinion is, that the sensory perception is the only source of insight. The next assumption to this topic wanted to include both approaches and said it is a combination of both. Till now there is only an agreement of the fact, that knowledge is not static and not the absolut truth.

*(vgl. Wissensmanagement in der Schulentwicklung-Theoretische Analyse und empirische Exploratio aus systematischer Sicht - Definitionsproblematik, Kaja Heitmann, 2012)*

## 8.2. Knowledge management in context of the knowlegde socitey

Knowledge management is inseparable from the knowledge society because it is a response to the needs of the knowledge society. This explains why the talk about knowledge management should be preceded by an outline of the knowledge society. As early as the 1960s and 1970s, scientists described the change to a society in which knowledge becomes increasingly important and is added to the original production factors of an industrial society. In 1966, for example, the American sociologist Robert Lane was the first to speak of a "knowledgeable society" in an article in the American Sociological Review. In his 1969 work "The age of discontinuity", the American economist Peter Ferdinand Drucker outlined social changes within which the resource knowledge plays an important role. The popularization of the concept of a society based on knowledge ultimately goes back to the American sociologist Daniel Bell and his 1973 work "The coming of post-industrial society". However, the theory of the knowledge society should not be confused with its predecessors, such as the theory of the post-industrial society, because it does not share their world view of 'scientific faith'.

## 8.3. Knowledge management models

## 8.4. Knowledge management methods and tools

### 8.4.1. Repertory-Grid-Technique

This technique is used to acquire knowledge and represent it graphically. People often use mental models to solve problems in their everyday life. These models can be collected and presented



transparent with this technique.

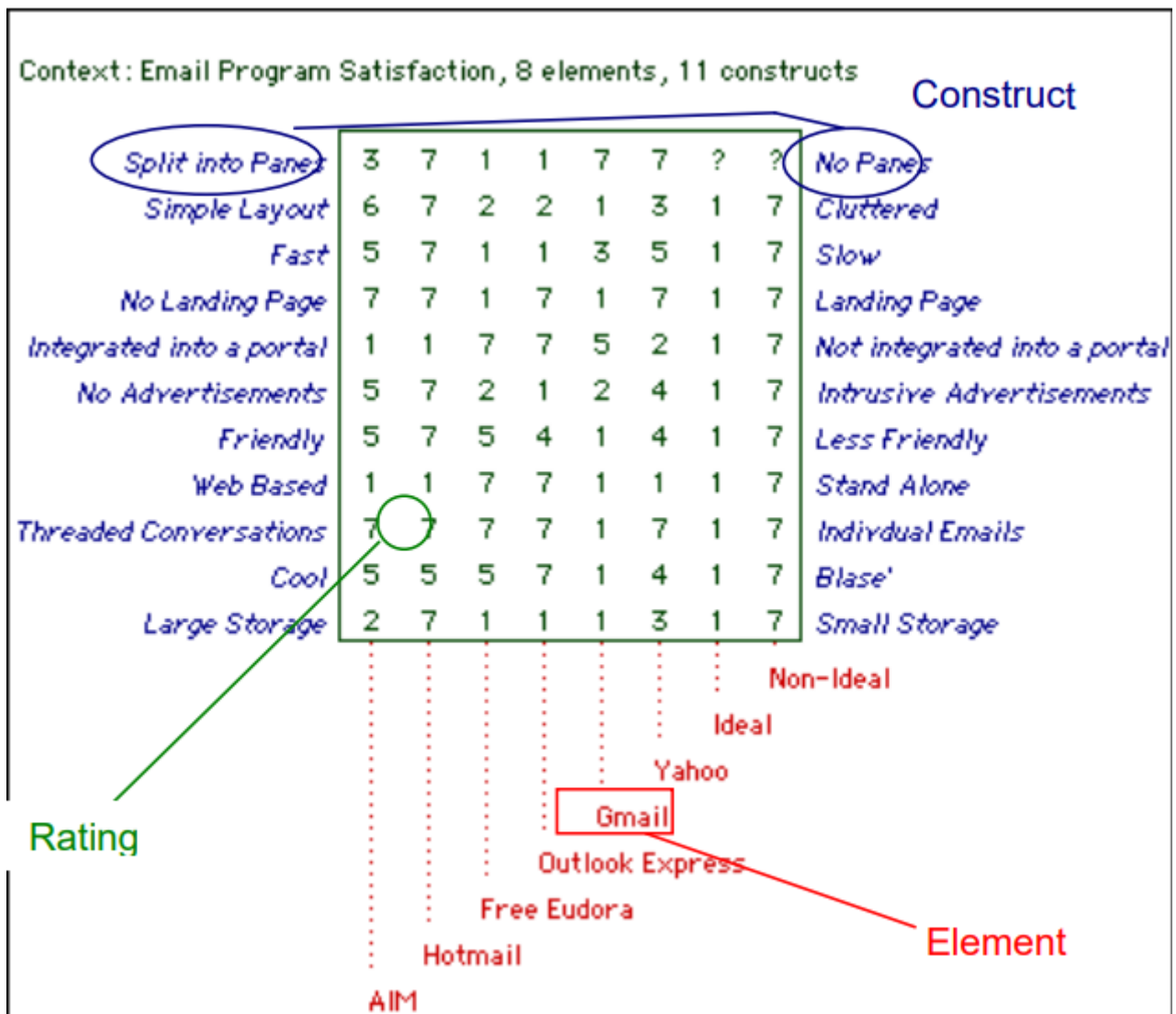


Figure 2. Sample Repertory Grid

Quelle ([https://www.researchgate.net/figure/Sample-Repertory-Grid\\_fig1\\_228296401](https://www.researchgate.net/figure/Sample-Repertory-Grid_fig1_228296401); letzter Zugriff 15.03.2020)

## Objective and possible applications

The aim of the method is to collect and present person-related views, implicit knowledge and underlying values on different areas. The RepertoryGrid technique helps to make individual mental models conscious and thus communicable to others, if used correctly and planned accordingly. Thus, the technology can be used in organisations to collect individual views of employees or other stakeholders on certain areas of knowledge and to either transfer this acquired knowledge into technical systems or to pass it on to other people. Possible fields of application of repertory grids today range from marketing to product development, from requirements surveys for technical systems to human resource management. In principle, the method is actually adaptable for all areas of knowledge. In particular, it can also be used to make changes visible, for example, if you want to find out and "measure" how a merger has changed the organizational culture (organizational level), or if you want to show which individual changes in personal knowledge or values have actually been achieved among the participants as a result of a personnel development measure (individual level), or if changes in customer needs are to be surveyed over time

(stakeholder level). The method allows those responsible to put themselves in the shoes of those affected, and to do so in a way that is specific to the topic or issue at hand. In addition, a side effect of the use of the method, which can also be the conscious goal of the survey, is that the type of questioning technique can trigger reflection processes related to knowledge. Thus the method can also be used as an intervention method for initiating individual or even organisational changes and learning. When exactly can the method be used? Let us assume, for example, that a clothing chain has difficulties in making its shops attractive to its target audience. Up to now, those responsible for the organisation have assumed that a target-group-oriented placement of products in the shop windows, in our case up-and-coming female managers, is the necessary attractor for high customer frequency. Now that the expected customer frequency has not materialised and customer surveys have not revealed any potential for improvement, repertory grid technology has been used.

## **Realisation**

The level of detail and adequacy of the results of the method is strongly dependent on the planning and moderation of the survey process and thus also on the moderation skills of the interviewer. The most important prerequisite for the correct implementation of the method is the observance of the rules of optimal interview management (the basics of this go back to Carl R. Rogers). For the interviewer, this means communicating as genuinely and unadulterated (congruent) as possible and showing appreciation and empathy for the interviewee. Consequently, the interviewer should openly engage with the contents of the interviewee's message and accept the counterpart as an independent personality with his or her own individual mental models. This attitude should enable individual answers that not only confirm the interviewer's mental models but can also contradict or complement them.

Empathy with the other person requires openness and a certain degree of honest curiosity about the other person. Establishing an empathic dialogue relationship promotes the willingness to talk and the actual acquisition of hidden knowledge content. Furthermore, the choice of the right question (no suggestive questions, such as "Don't you agree that grey is more beautiful than black?"), clarification of purpose and procedure prior to the actual interview and the creation of pleasant spatial and temporal conditions are prerequisites for a successful conversation.

If the interviewee is given enough time to think, the repertory grid method can also be used to depict constructs that are otherwise difficult to grasp, such as a feeling of discomfort when entering a business premises. These constructs are feelings and impressions that are usually not easily expressed in a word or a sentence. Here it is recommended to ask step by step (laddering in the sense of "What exactly is it that makes the two shops similar? How can you recognize this? What does this similarity have to do with the subject of the questioning?") until the implicit construct can be described. Only by adhering to these basic attitudes and considerations can a fruitful interview be conducted.

Implicit knowledge is above all context and situation dependent. During the survey, reference should be made above all to situations experienced by the interviewees, which the interviewees experienced as directly as possible before the interview. This makes it easier to remember. In the case of shop attractiveness, for example, a purchase that has taken place should be in the mind's eye of the respondents when they name the characteristics.

In an organisational context, it should not be forgotten that people with work experience or experience in a specific field of activity may not be motivated to share their previously unexplained

knowledge with others, as it is precisely this knowledge that sets them apart from other people in an organisation or may even make them indispensable. They may therefore refuse to participate in the implementation of the Repertory Grids or may not be particularly cooperative. Well-prepared information events and intensive individual discussions in advance are therefore particularly important. They can allay fears of losing supremacy and help raise awareness of the potential of the method.

*(vgl. Wissensmanagement in der Praxis-Zielsetzung und Einsatzmöglichkeiten, Christian Stary, Monika Moschner, Edith Stary, 2013)*

#### **8.4.2. Critical-Incident-Technique (CIT)**

The critical-incident technique allows the collection of observed behaviour that led to particular success or failure (this behaviour is therefore referred to as "critical" in the following) in performing a particular activity. Critical incidents are collected by means of questionnaires or interviews either with the performers of the successful or failed task themselves or with observers of the performance of tasks. The method takes particular account of the circumstances that led to the event, i.e. activities and factors that made the event successful or unsuccessful. It thus leads to specific descriptions of behaviour. The collection of critical events of an activity sharpens the awareness of the actions taken. In some cases the interviewees only become aware of the existence of these events during the survey, as they have not yet or only partially dealt with them beforehand.

*(vgl. Wissensmanagement in der Praxis-Critical Incident Technik, Christian Stary, Monika Moschner, Edith Stary, 2013)*

#### **Origin**

The origin of the critical incident technique goes back to Sir Francis Galton, who conducted studies in this direction as early as the end of the 19th century. Subsequently, controlled observation tests were developed, studies on recreational activities were conducted and anecdotal records were kept. The foundation of the Critical Incident Technique in its present form was not laid until the middle of the Second World War. In the summer of 1941, John Flanagan conducted studies as part of a US Air Force flight psychology program. The purpose of the program was to develop a procedure for selecting and classifying crews. The Critical Incident Technique has since been steadily developed and is now used in a variety of other areas.

#### **Objective and possible applications**

Critical Incident Technique aims to detect extraordinarily successful or unsuccessful work behaviour caused by critical events. Since the underlying behaviour is analysed, implicit (expert) knowledge is to be recorded and collected. Critical Incident Technique aims to improve work processes and help to avoid experienced errors in the future. Its application should make it easier for employees or those carrying out work activities to perform their tasks more effectively or more easily in the future. The method can also support those responsible in their decision-making in many areas, for example when hiring new employees. Typical issues where the Critical Incident Technique can be helpful are What qualifications should new employees have? How can I increase the motivation and productivity of my employees? How can frequently made mistakes be avoided in the future? On the basis of observed facts, the technology helps to find conclusive answers to

such questions and thus to develop options for action. The critical incident technique is particularly suitable for surveys in which a structured, behavioural method is required to raise knowledge or to make explicit knowledge transparent in a structured form and thus accessible to a new group of users. The technique has already been successfully applied in the following areas:

- **Military:** The method was introduced by John Flanagan during the Second World War in order to identify and process critical situations in aviation. Concrete events of effective and ineffective behaviour in aviation during the war should be found. To this end, he asked war veterans about events that were particularly important, helpful or inadequate for them to carry out the missions they had been assigned. One question for obtaining these descriptions of behaviour was for example: "Describe the officer's action. What exactly did he do?"
- **Police:** In this context, the method was used to analyse the activities of police officers in specific work situations. The relationship between stressful circumstances and certain behavioural patterns could be investigated using this technique. **Sales:** Analogous to the police, work situations were investigated which were characterised by certain customer-product constellations on the one hand and certain behavioural patterns on the other hand.
- **(Software) development:** In the context of this assignment, the coordination of tasks between managers and employees was the focus of interest.
- **customer service:** In this assignment, services were analyzed in detail from the consumer's perspective.
- **Housework:** In this context, the method was used to analyze conflicts that arise when couples with professional careers divide up their housework.
- **Training - concept development:** In this area, the method was used to develop definitions and theories of leadership and professionalism in order to impart knowledge.

Flanagan (1954) himself pointed out a number of other areas of application:

- **Measurement of typical performance criteria:** In this context, the critical incident technique was used to create an observation protocol list that included all the important courses of action for an activity. This list can then be used to objectively evaluate a person's performance.
- **Measurement of skills/knowledge (standard samples):** Standard samples were used to assess the knowledge of people concerning important aspects of their activities. This form is often used at the end of training courses to assess whether students have retained the knowledge they have acquired or can apply it correctly.
- **Teaching:** Many applications of critical incident techniques to problems in training have been developed for specific situations in the military. The technique is intended to help create better conditions for teaching, for example by strengthening motivating didactic moments.
- **Job design:** For a long time, insufficient attention was paid to job design, although it is essential to promote the motivation of individuals. In this area, the critical-incident technique attempts to limit the number of critical job elements of employees to two or three critical elements. This is intended to maximise the effectiveness of performance in relation to each of the different types of tasks.
- **Operating procedures:** Another application of the method is the study of operating procedures. The method helps to efficiently collect detailed, factual data based on successes or failures that can be systematically analysed. This is an essential prerequisite for improving the effectiveness

and performance of operating procedures.

- **Equipment design:** Here, the design of equipment or fittings is to be improved by collecting critical events in the handling of operating resources and tools. Reports "from the field" form the basis for improvements. Critical-incident technology facilitates the collection and processing of information to improve equipment and tools.
- **Motivation and leadership:** Critical incident technology was used in this context to collect data on specific actions, including decisions made and options chosen. From these data, causal relationships between work actions and leadership activities could be derived.
- **Psychotherapy:** The method is also used in this field. It serves as an aid in the collection of professional-critical events, with particular attention being paid to the interrelation of factors.

The Critical Incident Technique can therefore be used in many economic and social areas due to its openness in terms of content.

(vgl. *Wissensmanagement in der Praxis-Critical Incident Technik-Zielsetzung und Einsatzmöglichkeiten*, Christian Stary, Monika Moschner, Edith Stary, 2013)

## **Realisation**

The use of the critical incident technique proceeds along different phases, as described below. According to Flanagan, however, the definition of a critical event by the critical incident technique can only be considered valid and comprehensive if the observations are representative, if the persons performing the observations are sufficiently qualified, if the types of assessment are appropriate, if the steps used are suitable for generating accurate reports. Decisive for the quality of the surveys and thus the results is how the interview is conducted or how the questions are developed. Thus, one of the most important prerequisites for the correct implementation of the method (as with the methods "Repertory Grid" and "Narrative Storytelling") is the observance of the rules of optimal interview conduct by the interviewers. This requires the interviewer both to openly engage with the content that the interview partners are communicating and to accept the interview partners as the persons they represent. Two aspects are of particular importance for the successful application of the critical incident technique:

### **1. Questions**

They are the decisive aspect for data collection. Several studies have shown that a small change in the formulation of questions leads to a significant change in the results. For example, the same question was asked in two different ways. Since the respondents perceive or interpret questions differently, different answers can therefore be expected. For this reason, questions should first be asked to a smaller group of respondents before they are finally used. Misunderstandings can be avoided right from the start by determining whether the questions were formulated in a targeted manner and whether relevant answers can be expected. If the interviewers do not receive the desired quality of answers, the questions need to be reworked so that these problems do not occur again when the survey is finally conducted. The interviewers should all understand the same thing about the question. In addition, care should be taken to ensure that the interviewees describe the events or behaviour in a clearly defined situation and do not make (possibly extravagant) statements that deviate from the subject of the survey.

### **2. Interview conduct**

It is important that the interviewers behave neutrally. As soon as the respondents have given an answer, they should not be confused by inappropriate questions, for example by comments like: "Oh come on, is that really your opinion? Wasn't that completely different?" This could make respondents uncertain about their answers and the results obtained would lose quality and spontaneity. The respondents should be at the floor most of the time during the survey, whereas the interviewers should listen most of the time and only clarify possible problems of understanding of answers or questions. It is advantageous if the respondents are not interrupted. Instead, they should be given the feeling that their opinion as experts is accepted without reservation. If the interviewers get the impression that answers are not complete, they should ask the respondents to refine the respective answer, i.e. to expand on it, but not to correct it. This will motivate the respondents to mention as many details as possible.

### **8.4.3. Balanced Scorecard (BSC)**

#### **General**

The BSC is a method for the development and organisation-wide communication of an organisation's mission, vision and strategies derived from them. It can be described as a management system for the strategic management of an organisation with key figures. It is presented by means of a clearly arranged report sheet which contains not only results but also actions with which organisations prepare future activities. Furthermore, the results and actions are considered from different perspectives and in a balanced manner. Different types of BSCs are used in organisational practice. What these approaches have in common is that strategies are translated into concrete actions.

BSCs initially contain the formulation of a central strategic goal (key objective or vision) and the corresponding concretization of the key objective through sub-goals. The sub-goals are derived from several elements: Strategic orientations (topics or factors critical to success). Expectations of various stakeholders (= perspectives) regarding organizational potential. These are: Customers, business processes that primarily have an after-effect, employees (learning and development, innovation), finance and controlling, partners or competitors (suppliers, cooperation partners, associations etc.). The financial management is the focus of attention. The utilization of financial capital is definitely seen as an organization's ultimate goal. Therefore, the financial perspective represents the top level of a hierarchically structured BSC. This perspective is followed by the customer perspective, which describes the value proposition that is made available to the market. Below this is the perspective of the internal business processes, which comprise the value chain of the organization. This chain includes all activities necessary to create the value proposition for customers and transform it into growth and profitability for the shareholder. The foundation of the three perspectives is the learning and development perspective, as it defines intangible assets that are needed to take entrepreneurial activities and customer relationships to a higher level. The other elements of the Balanced Scorecard are: defined metrics as measures for key objectives and selected sub-objectives (strategic themes, perspectives), derived actions that meet the sub-objectives, defined metrics for the actions, organization of joint work for the practical implementation of the strategy (projects, action programs), integration of the metrics into the reporting system.

*(vgl. Wissensmanagement in der Praxis-Balanced Scorecard, Christian Stary, Monika Moschner, Edith Stary, 2013)*

## **Realisation**

Based on experience gained from relevant projects, the following prerequisites apply for the successful implementation of a BSC:

### **Teamwork**

A team that works well together and is able to communicate with each other produces higher quality results than individual workers. The team allows for mutual consideration of know-how and favours the motivation of "comrades-in-arms".

### **Top-down vs. bottom-up approach**

Work on the BSC begins with a joint definition of the organisational mission and vision and the strategic goals that are aligned with them. This can only be determined by top management in cooperation with the following departments. Then the top management level must accompany the process of implementing the BSC throughout the entire organisation, follow it, steer it and make the continuous revision of strategic and also operational goals its own task. However, the employees must be involved in this process, especially since the operational business is to be determined by them in any case (bottom-up design of work processes).

### **Mission statement (mission) and division of vision**

The management is obliged to state the mission and vision of its organisation in two or three understandable answers to the questions "How and what do we want to be seen as in the public eye?" and "Where do we want to be in five or ten years' time (vision)?" if it wants to make the setting and refinement of objectives transparent. Otherwise, there is a risk of misinterpretation and conflicts of understanding.

### **Strategy incorporation**

Not only the mission statement and the vision must be communicated to the employees, but also the strategies for the organisation developed from them. They must be known internally in such a way that all employees in the organization understand them as objectives for their daily operative work.

### **Realistic goal setting**

The basic principle of motivation to participate is that goals are presented in a comprehensible way and can be achieved with great effort. Tactically, it should be possible to achieve such ambitious goals in several stages, via milestones.

### **Use of exclusively strategically oriented key figures**

Goal setting alone is not enough - the actual and the target, i.e. the achievement of goals, must be measured. Only then can the person(s) responsible for the achievement of the objectives determine the current situation and inform the employees accordingly. For this reason, the BSC's key figures should only measure what represents the goal, namely the implementation of the strategy.

### **Minimum scope**

The "right" key figures should be developed by top management in cooperation with the BSC team. It is not the quantity but the quality of the key figures that is decisive. The latter not only enables concise statements with a high degree of detail, but also facilitates the handling of key

figures.

### **Linking key figures with responsibility**

Conclusions should be drawn from measured target achievement rates. Therefore, for each key figure, what needs to be done to achieve the goal must be determined and responsibilities for this goal achievement must be defined.

### **Trust as a control and management instrument**

The proximity of top management should be used to find out whether the right strategies are being measured with suitable key figures. The discussion within the organization should therefore be kept open, both to customers and suppliers and within the employees. In addition, a BSC must also take into account the dynamics of change, whether outside or inside the organization, which have consequences for key performance indicators. This includes the ability of organisations to deal with feedback and to derive learning processes from feedback, as well as the ability to cultivate communication and build trust within the organisation (Friedag, Schmidt 2001).

### **Linking the BSC of the organisational levels with the BSC of the organisation as a whole**

Once all employees are aware of the organisational strategy and involved in their activities, they should also be measured by the implementation of the strategies in their area of responsibility. Each area, each department should participate in the strategy and have its own key figures, and thus its own BSC. Practicability and comprehensibility: In this context, the BSC should fit on one page - visual representations of the achievement of objectives are also permitted.

### **Change management**

It is essential to deal with key figures that deviate from the plan. The implementation of strategies in the organisation should be the subject of a monthly reflection. This requires an institutionalisation of quality assurance processes, which can be carried out within the framework of certification.

### **Authenticity**

Since no BSC is like any other, an organisation must also go its own way in the further development of the BSC - it is mostly oriented towards the strengths, but also towards the identified weaknesses of its own organisation.

*(vgl. Wissensmanagement in der Praxis-Balanced Scorecard-Umsetzung, Christian Stary, Monika Moschner, Edith Stary, 2013)*



# 9. System, Scope and Context

System scope and context delimits the system from all its communication partners (neighboring systems and users). It thereby specifies the external interfaces.

## 9.1. Technical Context

The technical context describes interfaces linking the system to its environment.

The abstract technical context is shown by the following C4-Context Model:

### System Context diagram for the learning platform "GAME"

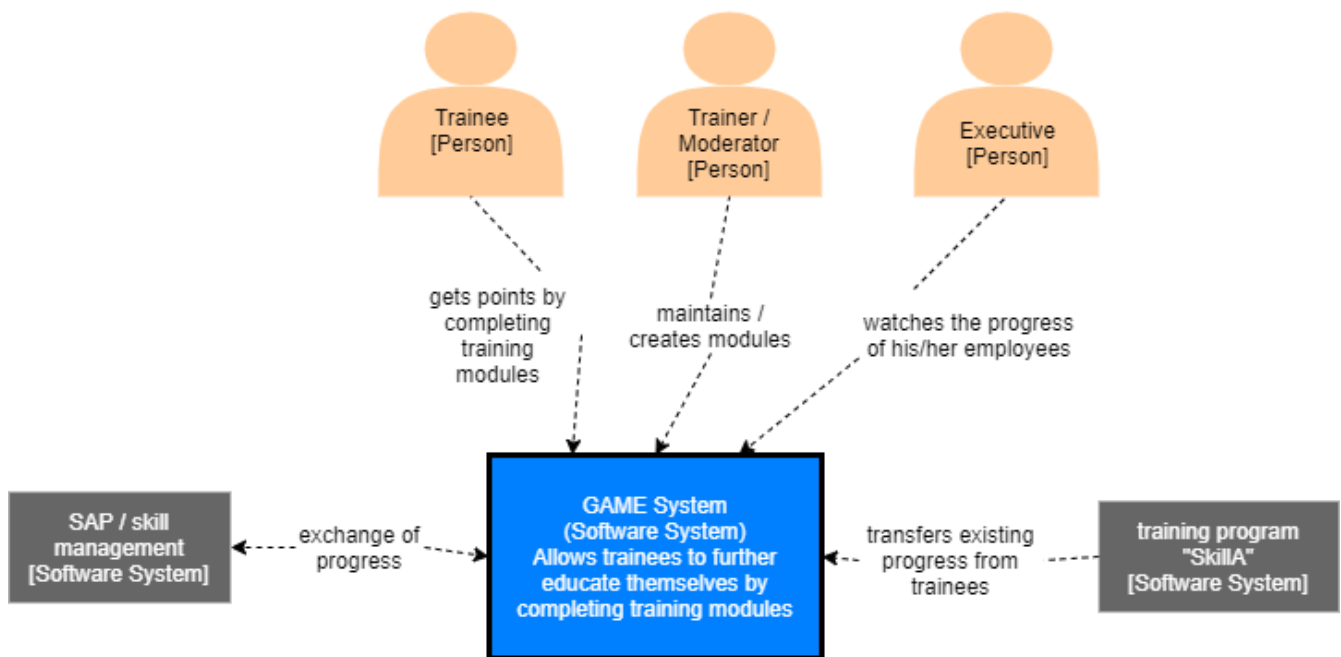


Figure 3. C4-Context Model

The GAME system is precisely described through this C4-Container Model:

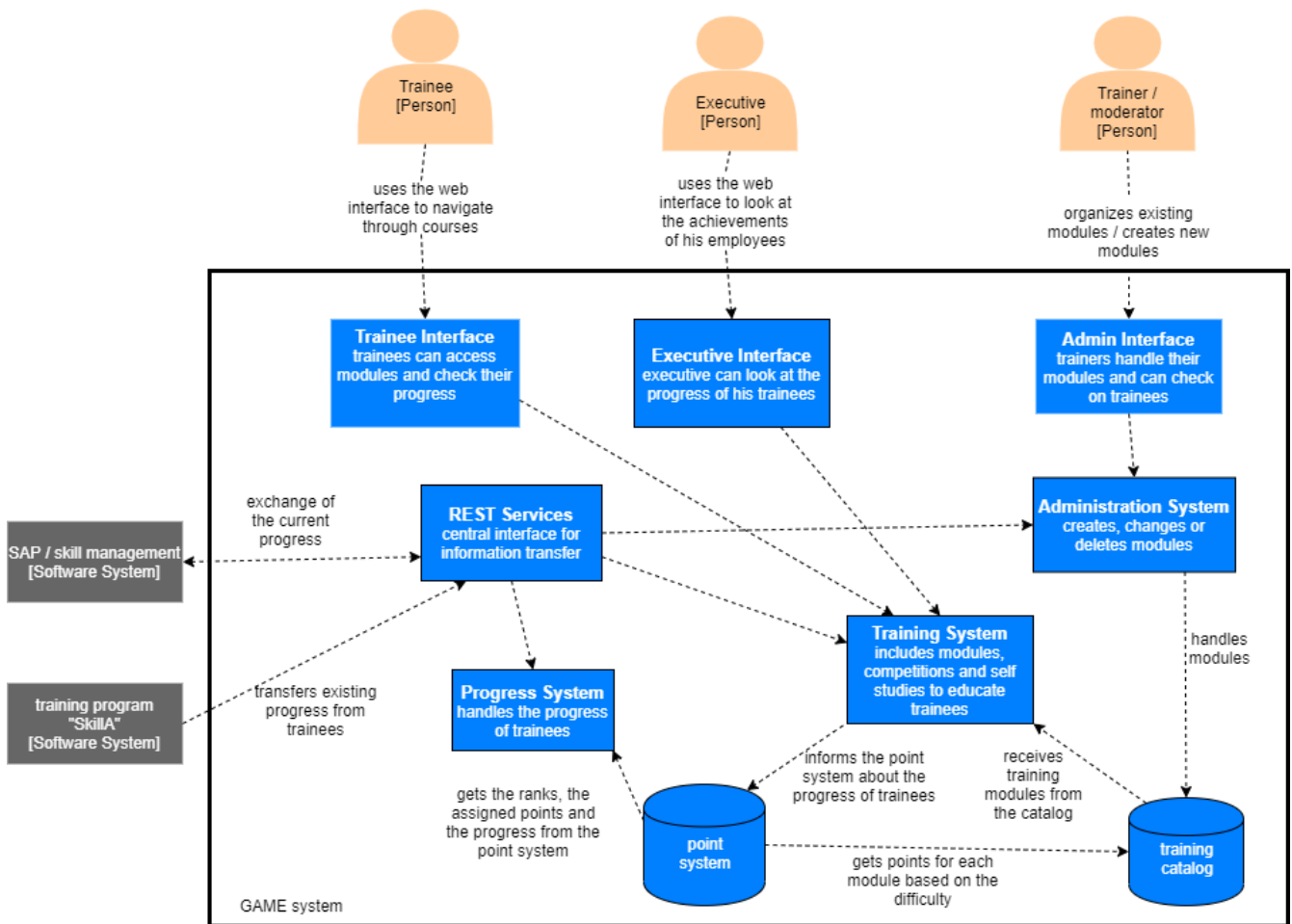


Figure 4. C4-Container Model

## 9.2. Description

The game system communicates with the following systems:

- Skill / SAP employee management system
  - This system transmits the current progress of the employees to the GAME System and vice versa.
- training system "SkillA"
  - This system transfers progress from the "SkillA" training system to the GAME system.

## 9.3. Considerations

In the point system several difficulty levels are defined which reward the user with previously determined points.

The trainer has to give every module he created a certain difficulty level.

In the beginning it should be possible to manually enter the progresses of trainees from "SkillA" in the progress system.

# 10. Solution Strategy

A short summary and explanation of the fundamental decisions and solution strategies, that shape the system's architecture.

## 10.1. Building from Scratch

The key decision of this project is the settlement to program the system from scratch, since there is no suitable, open-source platform which can be used for our purpose. Which systems were examined are written in [Building from Scratch](#).

## 10.2. Programming Language

The programming language we will use throughout this project is the highly acknowledged Java programming language. For in depth view of technology decisions please have a look in [Design Decisions](#).

# 11. Design Decisions

This chapter includes all decisions, the diploma team had to make. The following decisions are written in the [Lightweight Architecture Decision Records](#)-format.

## 11.1. ADR001-BuildingFromScratch

### 11.1.1. Status

accepted

### 11.1.2. Context

The ARZ wants to create a learning platform for their members. The project team evaluated possibly alternatives to adapt from. An alternative needs to be able to run on-premise, be up to date with today's standards and preferably be open source. The evaluated systems are the following:

<http://www.javaranch.com>

- is based on "JForum" (a java forum)
- OpenSource
- Code is on SourceForge

-> doesn't meet the expectations

<http://www.skill-guru.com>

- no On-premise possible

-> therefore excluded

<http://www.betterprogrammer.com>

- no further development, outdated (java version 1.5/1.6)

-> therefore not effective

<http://www.javapassion.com>

- no possibility for extensions
- developed by an individual person

-> therefore no viable option

### **11.1.3. Decision**

None of the evaluated systems meet the requirements, so the system will be build up from scratch by the project team.

### **11.1.4. Consequences**

Development of a web application based on Java/Java Enterprise.

## 11.2. ADR002-Technology stack

### 11.2.1. Status

accpeted

### 11.2.2. Context

The GAME application consists of different subsystems. To make it possible, that these subsystems are maintainable in the context of the whole application, it is important that these subsystems are based on the same base technology stack.

### 11.2.3. Decision

The GAME application and its subsystems are developed with the help of the [Java EE](#) technology stack. The Java EE technology is a proven technology which is standardised and offers therefore highest future-proofness.

### 11.2.4. Consequences

The GAME application uses Java EE in version 8 and is developed as application server neutral as possible. Version 8 is the latest production ready version of the plattform and therefore it is the version of choice for developing Java EE applications.

## 11.3. ADR003-Build and dependency management

### 11.3.1. Status

accepted

### 11.3.2. Context

The GAME application consists of different subsystems. To make it possible, that these subsystems can be automatically built and bundled a build and dependency management tool is needed.

### 11.3.3. Decision

[Apache Maven](#) will be used for build and dependency management. This tool is well-proven and can be extended with many plugins. Its convention-over-configuration paradigm lets a developer focus on creating the software.

## 11.4. ADR004-UI technology stack

### 11.4.1. Status

accpeted

### 11.4.2. Context

The primary user interface of the GAME application will be web based. Therefore a web UI Toolkit is needed for an efficient development of the UI and seamless integration with the backend systems.

### 11.4.3. Decision

As described in [ADR002-Technology stack](#) Java EE is the base technology stack for the GAME application. The Java EE standard defines the JSF substandard as its UI Framework/Toolkit for web based applications. Therefore the web based UI of the GAME application will be developed with the JSF framework.

### 11.4.4. Consequences

For some parts of a modern web based UI, the JSF framework doesn't have components to develop the UI efficient. For these parts it is allowed to use components of the [Primefaces](#) UI library.



## 11.5. ADR005-Communication with distributed systems

### 11.5.1. Status

accepted

### 11.5.2. Context

The GAME application must be able to communicate with other distributed systems in both directions (inbound and outbound).

### 11.5.3. Decision

The GAME application will provide resources for other distributed systems via [RESTful webservices](#).

By using a stateless protocol ([HTTP](#)) and standard operations, RESTful systems aim for fast performance, reliability, and the ability to grow by reusing components that can be managed and updated without affecting the system as a whole, even while it is running. It is also evolving as the defacto standard for communication between distributed systems in the digital age.

### 11.5.4. Consequences

JAX-RS is the standard for creating RESTful Webservices in Java EE based applications and so it will be used in the GAME application.

## 11.6. ADR006-Database

### 11.6.1. Status

accepted

### 11.6.2. Context

For persistent storage of data in the GAME application a database system is needed.

### 11.6.3. Decision

The [PostgreSQL](#) database will be used as the default database system in the GAME application.

PostgreSQL comes with many features aimed to help developers build applications, administrators to protect data integrity and build fault-tolerant environments, and help you manage your data no matter how big or small the dataset. In addition to being free and open source, PostgreSQL is highly extensible. It's worth mentioning that PostgreSQL also supports many NoSQL features as well.

### 11.6.4. Consequences

PostgreSQL is defined as the default database of the application, but database accesses must be as SQL standard compliant as possible, so that the portability of the application is still warranted.

## 11.7. ADR007-Java EE application server

### 11.7.1. Status

accepted

### 11.7.2. Context

Applications which are implemented on top of the Java EE specification, which is defined as the base technology of the GAME application in the [link:ADR002-Technology stack.adoc\[ADR002-Technology stack\]](#), need an application server as runtime environment.

### 11.7.3. Decision

The [Wildfly application server](#) will be used as the default application server in the GAME application.

The Wildfly application is open source and has full support for the Java EE 8 specification. It is also widely used in the ARZ and therefore the medium of choice.

### 11.7.4. Consequences

Wildfly is defined as the default application server of the application, but the application must be as Java EE standard compliant as possible, so that the portability of the application is still warranted. Further it must be possible to run the application also on application servers which are just Java EE Web Profile compliant.

## 11.8. ADR008-Standard code formatter

### 11.8.1. Status

accpeted

### 11.8.2. Context

For seamless working together on a shared code base it is advantageous to use the same code formatter. So a standard code formatter must be defined.

### 11.8.3. Decision

The standard code format is defined in the file [formatter.xml](#). This XML is created as an [eclipse](#)-Formatter file, but can also be importet in other IDEs, such as Apache Netbeans or IntelliJ Idea.

### 11.8.4. Consequences

Everyone who creates or changes source code for the GAME application in this repository, must use this formatter setup to format the source code.

# 12. Practical task: Eric Haneder (Front end)

## 12.1. Beginning

To start with designing user interfaces (= front end), a design must be agreed with the client which suits the client and is realizable for us. The front end is the most important part for the user, because it includes everything the user can see and interact with. I discussed this topic with the client and we came to a conclusion

Here is an abstract picture, how the user interfaces should look like:

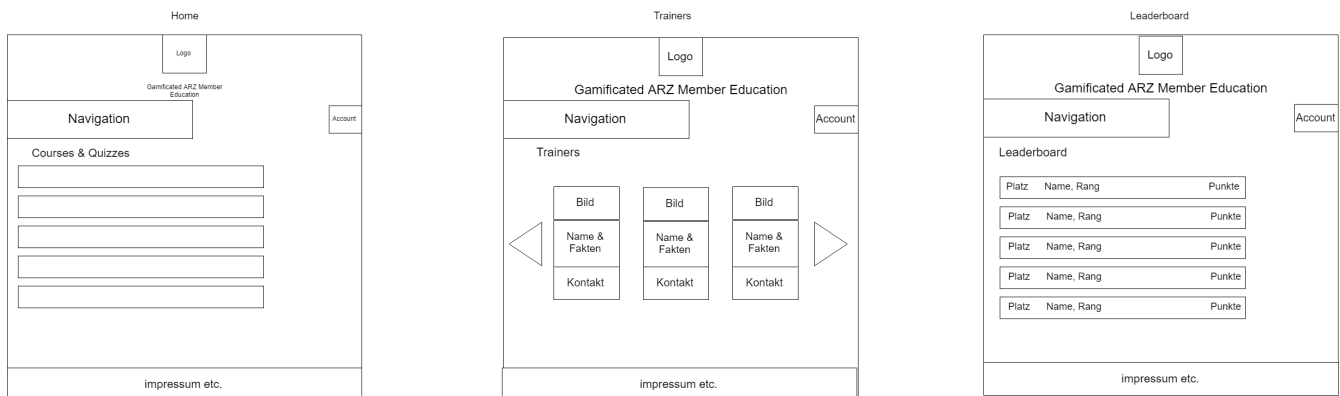


Figure 5. User interfaces

After the design was agreed upon, the programming of the interfaces could begin.

## 12.2. Fundamentals

To program a website, you need a lot of extensive knowledge. You have to know HTML, Javascript and CSS. In addition, you should also know how a finished page is generated from the code you programmed and how requests from the user can be evaluated and responded to.

### 12.2.1. JSF

Java Server Faces (JSF) is a programming framework for the development of graphical user interfaces (GUIs). JSF is a part of the web-technologies of *Java Enterprise Edition* (Java EE). The following graphic shows the architecture of JSF:

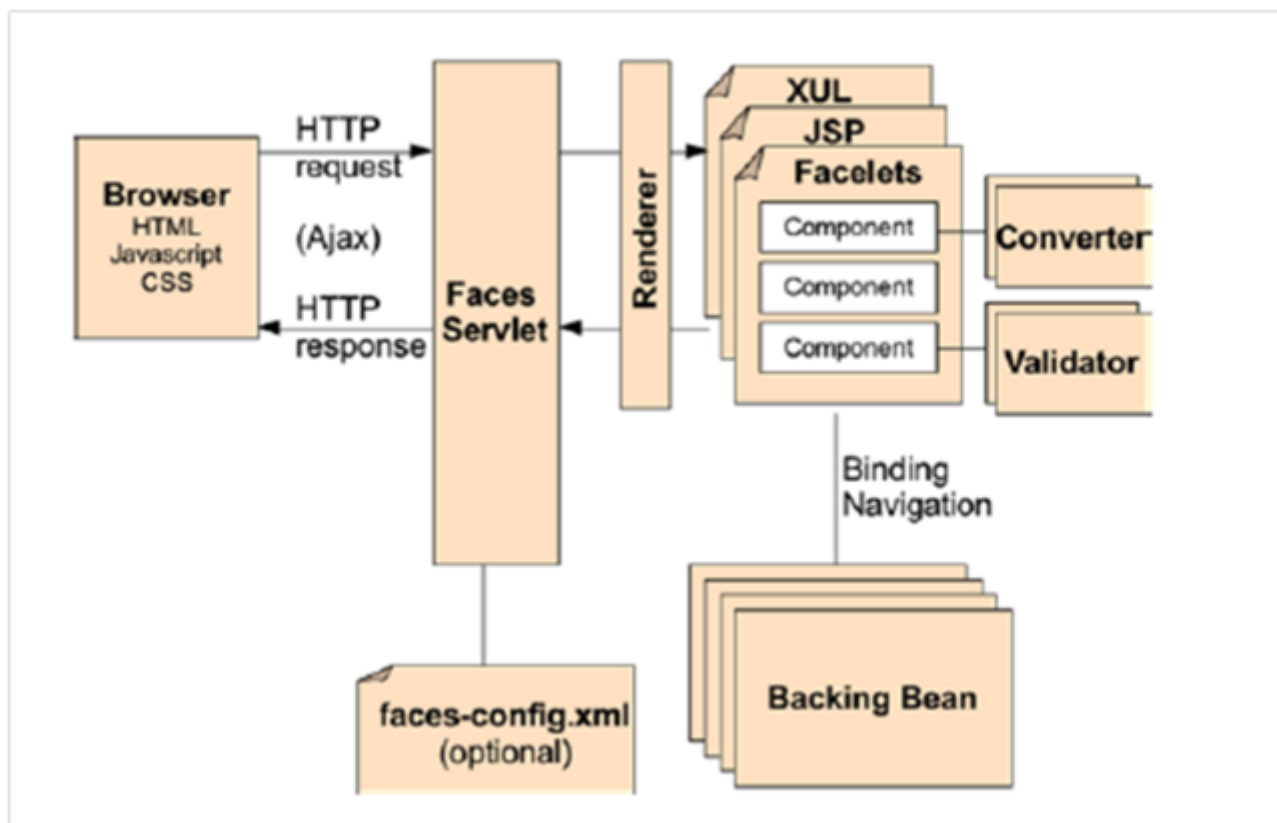


Figure 6. JSF architecture

## Browser

The Browser is there, to display the websites to the user. Through the browser, users can navigate through the interfaces, make requests to the server and get a response back.

## Faces Servlet

The Faces Servlet handles user interactions that may lead to changes in the data structure (back end) or on the user interfaces (front end). It can be seen as a controller between the front end and the back end. Every request runs through the Faces Servlet which takes action accordingly. It can optionally be configured by a *faces-config.xml* file.

## Renderer

The Renderers are responsible for displaying a component and translating a user's input into the component property values.

## Converter

Converters convert a component's value (Integer, Boolean, etc.) to and from markup values (String).

## Validators

Validators are responsible for ensuring that the value entered by a user is valid.

## Backing Beans

The business logic is made in backing beans, which also control the navigation between pages.

## Facelets

Facelets describes the language in which JSF files are written (XHTML). Every page has certain

components which will be displayed by the browser with the help of renderers.

(vgl. Goncalves A. (2013). *Beginning Java EE 7*. Apress.)

### 12.2.2. HTML

Hypertext Markup Language (HTML) is the common used language for building web pages. A HTML page is a text document (with a .html or .htm extension) used by browsers to present text and graphics. A web page is made of content, tags to change some aspects of the content, and external objects such as images, videos, JavaScript, or CSS files.

*Sample HTML code*

```
<html>
  <head>
    <title>My Webpage</title>
  </head>
  <body>
    <h1>This is my webpage</h1>
    <p>
      I hope you like it.
    </p>
    <a href="www.google.at">Link to Google</a>
  </body>
</html>
```



You can notice several tags in this code (such as <body> or <p>). Every tag has its own purpose, attributes and must be closed. Href is an attribute of the a-tag.

XHTML is just a validated version of html. This means, that there are certain rules, a html-page has to follow, to be valid. XHTML pages have a .xhtml extension.

Some of the rules are the following:

- All tags must be closed. (so no <br>, <hr>, ...)
- All tags are lowercase.
- Attributes appear between single or double quotes (<table border="0"> instead of <table border = 0>)
- There must be a strict structure with <html>, <head> and <body> tags.

### 12.2.3. CSS

Cascading Style Sheets (CSS) is a styling language used to describe the presentation of a document written in html or xhtml. CSS is used to define colors, fonts layouts, and other aspects of document presentation. It allows separation of a document's content (written in XHTML) from its presentation (written in CSS). To embed a .css file in your html or xhtml page, use the <link> tag. (e.g. <link rel="stylesheet" type="text/css" ?

```
p {  
    font-size: 10px;  
}  
h1 {  
    color: red;  
    font-style: italic;  
}
```

## 12.3. Getting Started

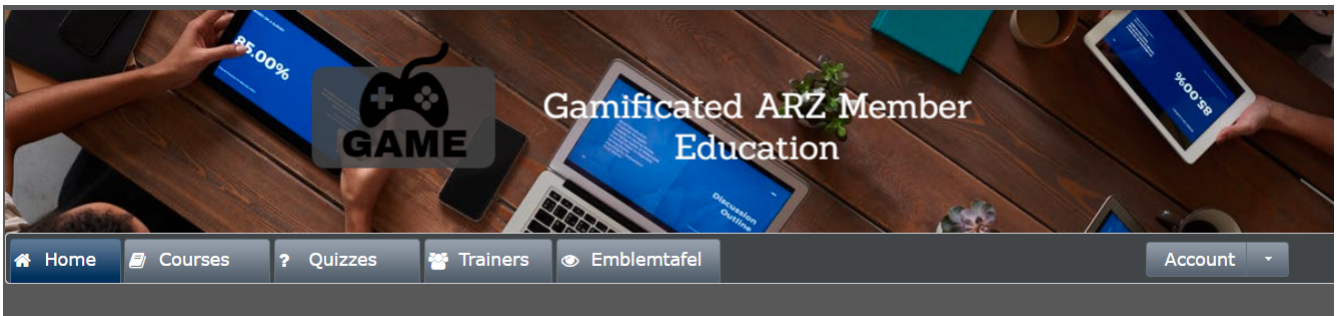
### 12.3.1. Creating a project structure

To start with a project like this, you need a clean project structure. The project structure has been automatically created with Maven. Our project is "Open Source", that means it is available online for free. That includes our whole project structure and all of the files used to create the website. The complete GAME project can be viewed under: <https://github.com/game-admin/game>

### 12.3.2. Creating a layout

To simplify the creation of the UIs, I prepared a layout which serves every page as a template. This layout is realised in "mainlyaout.xhtml". It constructs the picture at the the top of every file with the headline, the menubar and a little picture in the footer.

*Design/Layout of the game platform*



This page is not displayed directly to the user, but rather represents a template for all user interfaces. Other pages can define this page as a template and then adopt its content. By using `<ui:insert>` in the template file, you can give the template clients the possibility to define the content of this tag themselves with `<ui:define>`. This is shown in any of the actual user interfaces.

The menubar is constructed on another file called "menubar.xhtml". It is used to navigate through the platform. Furthermore, the menubar file is very unique, because it is not used directly in every file, but is displayed on every page. This is due to all of the pages using the mainlayout file as a template. It is include in the mainlayout file with the `<ui:include>` tag:

```
<ui:include src="./faces/menubar.xhtml"></ui:include>
```



The menubar is created by using the `<p:tabMenu>` tag of the Primefaces library. It is very convenient to create a menubar with this tag.

*menubar.xhtml*

```
<p:tabMenu style="width:100%">
  <p:menuitem value="Home" outcome="index.xhtml" style="width:5em" icon="fa fa-home"
">
  </p:menuitem>
  <p:menuitem value="Courses" outcome="courses.xhtml" style="width:7em" icon="fa fa-
book">
  </p:menuitem>
  <p:menuitem value="Quizzes" outcome="quizzes.xhtml" style="width:7em" icon="fa fa-
question">
  </p:menuitem>
  <p:menuitem value="Trainers" outcome="trainers.xhtml" style="width:6em" icon="fa
fa-users">
  </p:menuitem>
  <p:menuitem value="Emblemtafel" outcome="leaderboard.xhtml" style="width:8em"
icon="fa fa-eye">
  </p:menuitem>
</p:tabMenu>
```

The "Logout" button on the far left is done with the `<p:splitButton>` tag. It is placed in the menubar with CSS.

*menubar.xhtml*

```
<p:splitButton id="basic" value="Account" action="index.html">
  <p:menuitem value="Quizzes" action="quizzes.xhtml"/>
  <p:menuitem value="Courses" action="courses.xhtml"/>
  <p:separator/>
  <p:menuitem value="Logout" url="http://www.google.com"/>
</p:splitButton>
```

I had to separate the menubar from the main layout, because of interferences with the formatters.

## 12.4. User-Interfaces

User Interfaces describe everything, the user can use to communicate with the application. Die UIs liegen hier unter `src/main/webapp/faces`.

The index-page is the standard page the browser will run, if you enter a website. On this page, the user should get an overview about his statistics, and he should be able to navigate to other pages. The structure of the index-page is pretty simple. All data of the current user is fetched from the database and displayed. The trainee can look at his nickname, score, progress and emblems. Furthermore, a little description of the GAME-site is displayed. From here on, the trainee can check out some courses, take quizzes, look at trainers or visit the emblemboard.

The data is displayed via expression language. Here is an example:

```
#${traineeController.getTraineesByID("1").get(0).nickname}
```

### 12.4.1. Courses-page

The Courses-page should display a list of courses the trainee can go through. These courses can be mandatory to complete Quizzes.

*courses.xhtml*

```
<p:dataTable var="kurs" value="#{kursController.kurse}">
  <p:column headerText="Titel">
    <h:outputText value="#{kurs.titel}"></h:outputText>
  </p:column>
  <p:column headerText="Beschreibung">
    <h:outputText value="#{kurs.beschreibung}"></h:outputText>
  </p:column>
  <p:column>
    <h:form>
      <!--<p:commandButton
action="{kursController.takeKurs(kurs.kursID)}" value="Take Course!" >
      </p:commandButton>      -->
      <!--<h:commandLink
action="{kursController.kursbean.find(kurs.kursID).getLink()}" value="Take
Course!"></h:commandLink-->
      <h:commandLink action=
"https://www.tutorialspoint.com/java/index.htm" value="Take Course!"></h:commandLink>
    </h:form>
  </p:column>
</p:dataTable>
```

The Primefaces tag `<p:dataTable>` takes a list and knows how to display its content through the columns. It is the equivalent to the `<table>` tag of html, but with some extra functions and style modifications. Here, I put in a list of courses. Every course in the list has a title, description and a link which is displayed in separated columns.

The `<p:commandButton>` invokes the `takeKurs`-method when pressed. In this method, a link of the selected course is returned. The `<h:commandLink>` is linked to the URL of the course. By clicking on it, the user is redirected to the site of the course.

This is how the courses interface looks like:

Home	Courses	Quizzes	Trainers	Emblemtafel	Account
------	---------	---------	----------	-------------	---------

Kurse

Hier kannst du Kurse nehmen, die dich auf die Quizzes vorbereiten.  
Nachdem du dir einen Kurs angeschaut hast, wird das jeweilige Quiz freigeschalten!

Titel	Beschreibung	
Start-Kurs	Dieser Kurs ist der erste Kurs der abgelegt werden muss. Er erklärt dir die Basics von Java.	Take Course!
Object & Classes	In diesem Kurs geht es um Objekte und Klassen die in Java sehr wichtig sind. Der Kurs erklärt dir die Grundlagen, die du über dieses Thema wissen musst.	Take Course!
Constructors	Das ist ein Kurs über die Konstruktoren in Java. Konstruktoren sind wichtig um ordentlich mit Objekten und Klassen arbeiten zu können.	Take Course!

Figure 7. Courses page

## 12.4.2. Quiz pages

The quiz pages include a interface, where every takeable quiz is displayed, two pages for taking a quiz and a page where the results are shown. Every quiz has its own emblem, which can be won if they quiz is taken succesfully. This means the user has to has at least half of the questions right.

*quizzes.xhtml*

```
<p:dataTable var="quiz" value="#{quizController.quizzes}">
  <p:column headerText="Titel" rendered="#{quizController.isTakeable(quiz.QID,
  &quot;1&quot;)}">
    <h:outputText value="#{quiz.titel}"></h:outputText>
  </p:column>
  <p:column headerText="Beschreibung" rendered=
  "#{quizController.isTakeable(quiz.QID, &quot;1&quot;)}">
    <h:outputText value="#{quiz.beschreibung}"></h:outputText>
  </p:column>
  <p:column rendered="#{quizController.isTakeable(quiz.QID, &quot;1&quot;)}">
    <h:form>
      <h:commandButton action="#{quizController.quizUebergabe(quiz.QID)}" value
      ="Take Quiz!" >
        </h:commandButton>
      </h:form>
    </p:column>
</p:dataTable>
```

Here, `<p:dataTable>` is used again, this time to show all available quizzes. It gets a list of quizzes and displays a quiz whether or not it is takeable. This is evaluated in the `isTakeable`-method in the `QuizController` Bean.

This is how it looks like, when the user has not fulfilled the requirements to take a quiz:

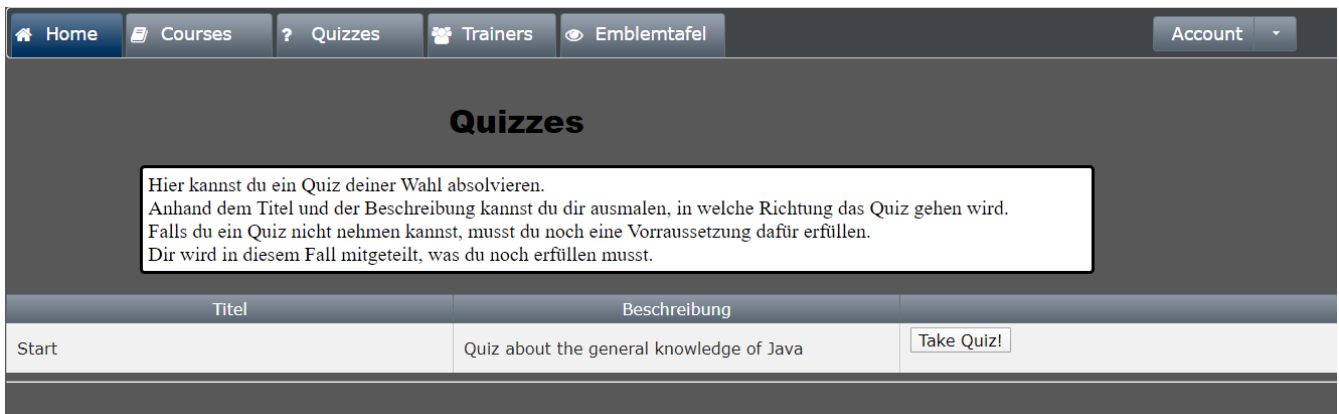


Figure 8. Quizzes page

In this picture below, the trainee has met all the requirements needed to take the other two quizzes. The second quiz requires the first first quiz to be completed succesfully and the third quiz requires the second quiz.

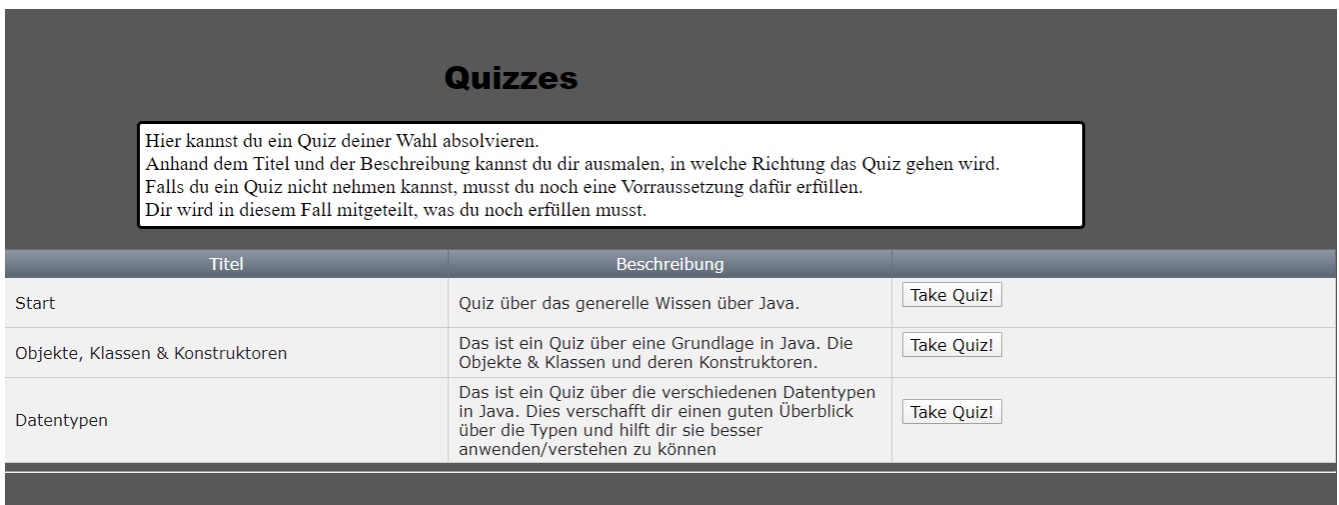


Figure 9. Quizzes page with fulfilled requirements

By clicking on the <p:commandButton>, the user can take the quiz.

```

<ui:repeat var="frage" value="#{quizController.fragemodell}">
  <div class="question">
    <h:outputLabel for="radio" value="#{frage.frage}"></h:outputLabel>
    <div class="answer">
      <p:selectOneRadio id="radio" value="#{frage.selectedAnswer}" layout="grid"
required="true" unselectable="true" columns="1">
        <f:selectItem itemValue="#{frage.antworten.get(0)}" itemLabel=
"#{frage.antworten.get(0)}"></f:selectItem>
        <f:selectItem itemValue="#{frage.antworten.get(1)}" itemLabel=
"#{frage.antworten.get(1)}"></f:selectItem>
        <f:selectItem itemValue="#{frage.antworten.get(2)}" itemLabel=
"#{frage.antworten.get(2)}"></f:selectItem>
        <f:selectItem itemValue="#{frage.antworten.get(3)}" itemLabel=
"#{frage.antworten.get(3)}"></f:selectItem>
      </p:selectOneRadio>
      <br/>
    </div>
  </div>
</ui:repeat>

```

This is the page for taking singlechoice-quizzes. The questions are repeatedly displayed by the `<ui:repeat>` tag. This tag runs through a given list, and displays the wanted data. The answers are displayed with the `<p:selectOneRadio>`, which renders a set of buttons based on the data you set with `<f:selectItem>`.

# Java Learning Quiz!

Für jede richtig beantwortete Frage bekommst du 10 Punkte!  
Es ist immer nur eine Antwort richtig!



Wenn du alle Fragen richtig beantwortest, bekommst du dieses Emblem:

What are advantages of Java?

- ☐ Flawless
- ☒ Platform Independent
- ☐ Only compatible with Windows
- ☐ only compatible with Linux

What is written after a line of code?

- ☐ "."
- ☐ "{ or } "
- ☒ ";"
- ☐ "</> "

Figure 10. Single-choice quiz

What is a feature of Java?

- ☐ High graphical visualisation
- ☐ Dynamic
- ☐ Platform Dependant
- ☐ Insecure

Check Answers

Figure 11. Bottom half of the single-choice quiz

By clicking on the "Check Answers" button, the trainee is redirected to the results page.


The page for taking multiplechoice quizzes looks almost the same, except for the buttons. I used `<p:selectBooleanCheckbox>` tags here, because they can be used for multiplechoice purposes. Each

of these buttons must be bound to a Boolean property.

## Java Learning Quiz!

Für jede richtig beantwortete Frage bekommst du 10 Punkte!  
Es können mehrere Antworten richtig sein!

Wenn du alle Fragen richtig beantwortest, bekommst du dieses Emblem:



How many primitive datatypes exist in Java?

6 ☐

7 ☐

8 ☒

9 ☐

What is/are a data type in Java?

int ☒

double ☒

String ☒

text ☐

Figure 12. Multiple-choice quiz

The results page is responsible for displaying the results of the quiz taken by the trainee. The trainee is able to see how many questions he/she answered right, how many points he/she won and which questions he/she answered incorrectly. Furthermore, the user sees which answers is right for every question. If the question is green and checked, the user answered correctly. If the question is red with a X at the end, the user answered incorrectly.

## Results

Du hast 8/10 Fragen richtig beantwortet!

Damit bekommst du 80 Punkte!

Die von dir richtig/falsch beantworteten Fragen siehst du hier mit den richtigen Antworten:

**What concept of the following is Java supporting? ✓**

Multiple Inheritance

Classes&Methods

Functions

None of the above

**Which things do objects include? ✗**

Classes

only state

only behaviour

state and behaviour

**What is not a variable type which classes can contain?**

✓

Global

Local

Class

Instance

Figure 13. Results page

### 12.4.3. Trainer page

The Trainers page should display all the trainers associated with the GAME platform. The trainees can contact these trainers if they need help.

*trainers.xhtml*

```
<p:carousel value="#{trainerController.trainers}" headerText="Trainers" var="trainer"
itemStyle="text-align:center" responsive="true">
  <p:panelGrid columns="2" style="width:100%;margin:10px 0px" columnClasses=
"label,value" layout="grid" styleClass="ui-panelgrid-blank">
    <f:facet name="header">
      <p:graphicImage library="img" name="trainer.jpg"/>
    </f:facet>
    <h:outputText value="Name:" />
    <h:outputText value="#{trainer.name}" />
    <h:outputText value="Rolle:" />
    <h:outputText value="#{trainer.role}" />
    <h:outputText value="Abteilung:" />
    <h:outputText value="#{trainer.branch}" />
  </p:panelGrid>
</p:carousel>
```

Here I used a primefaces tag called `<p:carousel>`. This tag is used to create a carousel. The `<p:panelGrid>` tag is used to display data in a grid. The `<p:graphicImage>` is just like the JSF tag `<h:graphicImage>`. `<h:outputText>` is used to display text, with the function to call a Backing Bean.



For easier explanation here is a picture:

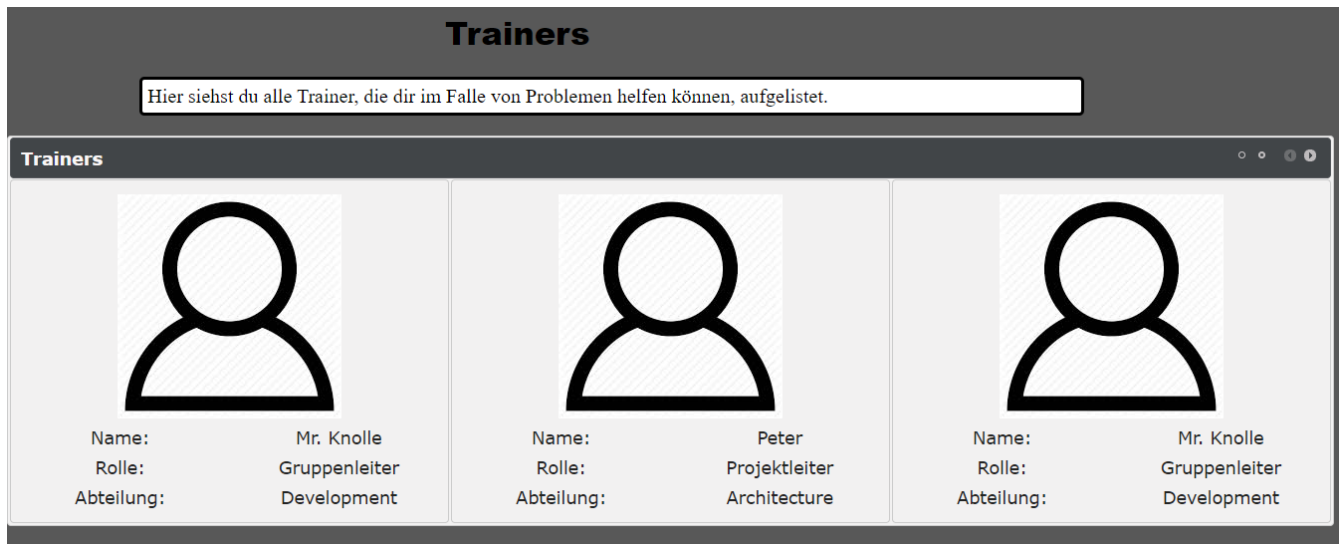


Figure 14. Trainers page

## 12.4.4. Emblemboard

The emblemboard page is used to display all the trainees with their names, nicknames, branches and Icons they got. You should be able to sort them by their names.

*leaderboard.xhtml*

```
<p:dataTable var="trainee" value="#{traineeController.trainees}">
  <p:column headerText="Name">
    <h:outputText value="#{trainee.vorname}" />
    <h:outputText value="#{trainee.nachname}" />
  </p:column>

  <p:column headerText="Nickname">
    <h:outputText value="#{trainee.nickname}" />
  </p:column>

  <p:column headerText="Abteilung">
    <h:outputText value="#{trainee.abteilung}" />
  </p:column>

  <p:column headerText="Embleme">
    <h:graphicImage value="data:image/png;base64,#{trainee.embleme}"
  ></h:graphicImage>
  </p:column>
</p:dataTable>
```

Here, the `<p:dataTable>` tag is used once again. This time it displays all the trainees with their emblems for the other trainees to see. Users can compare each other and are able to check out how the other users are doing.



Emblemtafel			
Hier siehst du eine Auflistung aller Trainees, mit ihren Nicknames und ihren Emblemen.			
Name	Nickname	Abteilung	Embleme
Eric Haneder	ericbensi	Softwaredevelopment	
Alex Saliger	SaAlexX_1010	Hausmeister	
Alexander Wurst	wursti	Front-End Development	
Jan Binder	Syreax	Projektmanagement	

Figure 15. Emblemtafel page

## 12.5. Java classes

Java Classes contain business logic that are need for the application. For example, if you want to display data on a page, you have to fetch data. This is done with Java files. To be more specific, Java classes which communicate with pages are called *Backing Beans*.

Backing Beans are identified by their `@Named` annotations. Furthermore, every bean has to have a scope annotated. An example for this is the `TrainerController` class I programmed.

*TrainerController.java*

```
@Named
@ViewScoped
public class TrainerController implements Serializable {

    private List<Trainer> trainers;

    private Trainer selectedTrainer;

    @Inject ①
    private TrainerService service;

    @PostConstruct
    public void init() {
        trainers = service.createTrainers(6);
    }

    //Getters & Setters
}
```

- ① Here, the `TrainerService` class is injected via the Java EE Dependency Injection System. This way, we can use everything from the injected Class, without the need of calling a constructor. The `TrainerService` generates a list of trainers to be displayed by the `trainers` page.

The `TrainerController` class is used to display all the trainers on the `trainers.xhtml` page. The list of trainees is used in the `<p:carousel>` tag.

Another class I developed is the `TraineeController`. This class is used to display all the trainees on the emblemboard. It looks quite similar to the `TrainerController`. The only difference is that the

data is created by the TraineeEJB class.

The last class I programmed, is the biggest one. It is called QuizController.java and it handles all of the interactions regarding the quizzes. This means it is responsible for forwarding the trainee from the quizzes page to the takquiz page, and from the takequiz page to the results page. Furthermore, it evaluates if a trainee meets all the requirements to take a quiz and it evaluates the results of a taken quiz.

### 12.5.1. Quiz Classes

*QuizController*

```
public void evaluateScoreMultiple() {
    List<Integer> falsche = new ArrayList<>();
    int richtige=0;
    for(int i=0; i<fragemodell.size(); i++ ) {
        List<Integer> indexrichtig = fragemodell.get(i).indexrichtig;
        for(int z=0; z<4; z++) {
            if(indexrichtig.get(z) == 1 && !fragemodell.get(i).buttons[z] ||
indexrichtig.get(z) == 0 && fragemodell.get(i).buttons[z]) {
                falsche.add(i);
                z=999;
            } else {
                richtige++;
            }
        }
        if(richtige==4) {
            score+=10;
            ricounter++;
            falsche.add(9999);
        }
        richtige = 0;
    }
    checkResults(falsche);
}
```

This method is used to evaluate the results of a MultipleChoice-Quiz. Each button is bound to a boolean-wert of the buttons[]. Every Question is checked, if every button matches the right answers. The user only gets points, if he answers the question correctly.

```

public void evaluateScoreRadio() {
    List<Integer> falsche = new ArrayList<>();
    //fragemodell = creator.createModell(qid);
    int indexri=0;
    for(int i=0; i<fragemodell.size(); i++) {
        for(int j=0; j<fragemodell.get(i).indexrichtig.size(); j++) {
            if(fragemodell.get(i).indexrichtig.get(j) == 1)
                indexri = j;
        }
        if(fragemodell.get(i).selectedAnswer.equals(fragemodell.get(i).antworten
.get(indexri))) {
            score+=10;
            ricounter++;
            falsche.add(9999);
        } else {
            falsche.add(i);
        }
    }
    checkResults(falsche);
}

```

Here, the singlechoice-quizzes get evaluated. This is much easier, because the radiobuttons function differently than the normal buttons. Every set of radiobuttons is bound to one value (selectedAnswer). We only need to check if the selected Answer matches the correct Answer.

```

public void checkResults(List<Integer> falsche) {
    results = new ArrayList<>();
    for(int i=0; i<fragemodell.size(); i++) {
        List<Integer> indexrichtig = fragemodell.get(i).indexrichtig;
        if(falsche.get(i) == i) {
            results.add(new Results(fragemodell.get(i).frage, fragemodell.get(i)
            .antworten, indexrichtig, true));
        } else {
            results.add(new Results(fragemodell.get(i).frage, fragemodell.get(i)
            .antworten, indexrichtig, false));
        }
    }
    trainee = traineebean.find("1");
    trainee.setProgress(trainee.getProgress()+score);
    traineebean.update(trainee);
    List<Quizbeantwortung> list = quizbeantw.findByQIDAndMITID(qid, "1");
    list.get(0).setErreichtePunkte(score);
    if(score > fragemodell.size()*10/2) {
        list.get(0).setIstbestanden(true);
    }
    quizbeantw.update(list.get(0));
}

```

In *checkResults*, a *Results-List* is generated. This list is used to display the results on the results.xhtml page.

The QuizController is used to handle everything surrounding the action of taking a quiz. It is responsible for displaying the content on the quizzes-, takequiz- and results-page. It will forward the user from the quizzes page to the takequizpage, where he/she can take the quiz. By clicking on the Submit button, the user is forwarded to the results-site, where their results are shown.

# 13. Practical task: Jan Binder (Back end)

## 13.1. Beginning

First of all the basics, like the technology we are using and the design are set. The database, in which all the tables and data sets are, is also appointed.

## 13.2. Fundamentals

You need some foreknowledge to build a database and a database connectivity. Knowledge about SQL, JPA and Java is required. You need these languages to create queries or to understand how the program communicates with the database.

### 13.2.1. SQL

SQL is short for "Structured Query Language". It is used to insert, change and delete data sets or create or delete tables.

*Beispielcode von SQL*

```
①
CREATE TABLE mitarbeiter (
    id integer,
    nachname text,
    vorname text,
    gehalt integer
);

②
INSERT INTO mitarbeiter (id, nachname, vorname, gehalt)
VALUES (1, Mustermann, Max, 2000);

③
SELECT * FROM mitarbeiter;

④
ALTER TABLE mitarbeiter ADD CONSTRAINT "MITARBEITER_pkey" PRIMARY KEY ("id");
```

- ① All this code does is it creates a new table called "mitarbeiter" which contains the columns "id", "nachname", "vorname" and "gehalt". Every column has its datatype next to it. Optionally there are constraints to the columns, such as "NOT NULL". which tells the database that this column cannot be empty or "UNIQUE", which means that a data set must be unique in this column.
- ② The next command is INSERT INTO. This code tells the database to write data into the table. First you need to set the table with all its columns and then the values like shown.
- ③ The "SELECT \* FROM mitarbeiter;" command picks out every data set from the table "mitarbeiter". This is done by the '\*' parameter. Further there is a possibility to only select a few

specific data sets.

- ④ This command lets you change an already existing table. In this example I updated the tables constraints, in detail I set the primary key of the table.

### 13.2.2. JPA

Applications are made up of business logic, interaction with other systems, user interfaces and data. Most of the data that our applications manipulate have to be stored in databases, retrieved, and analyzed. Databases are important: they store business data, act as a central point between applications, and process data through triggers or stored procedures. Persistent data are everywhere, and most of the time they use relational databases as the underlying persistence engine (as opposed to schemaless databases). Relational databases store data in tables made of rows and columns. Data are identified by primary keys, which are special columns with uniqueness constraints and, sometimes, indexes. The relationships between tables use foreign keys and join tables with integrity constraints.

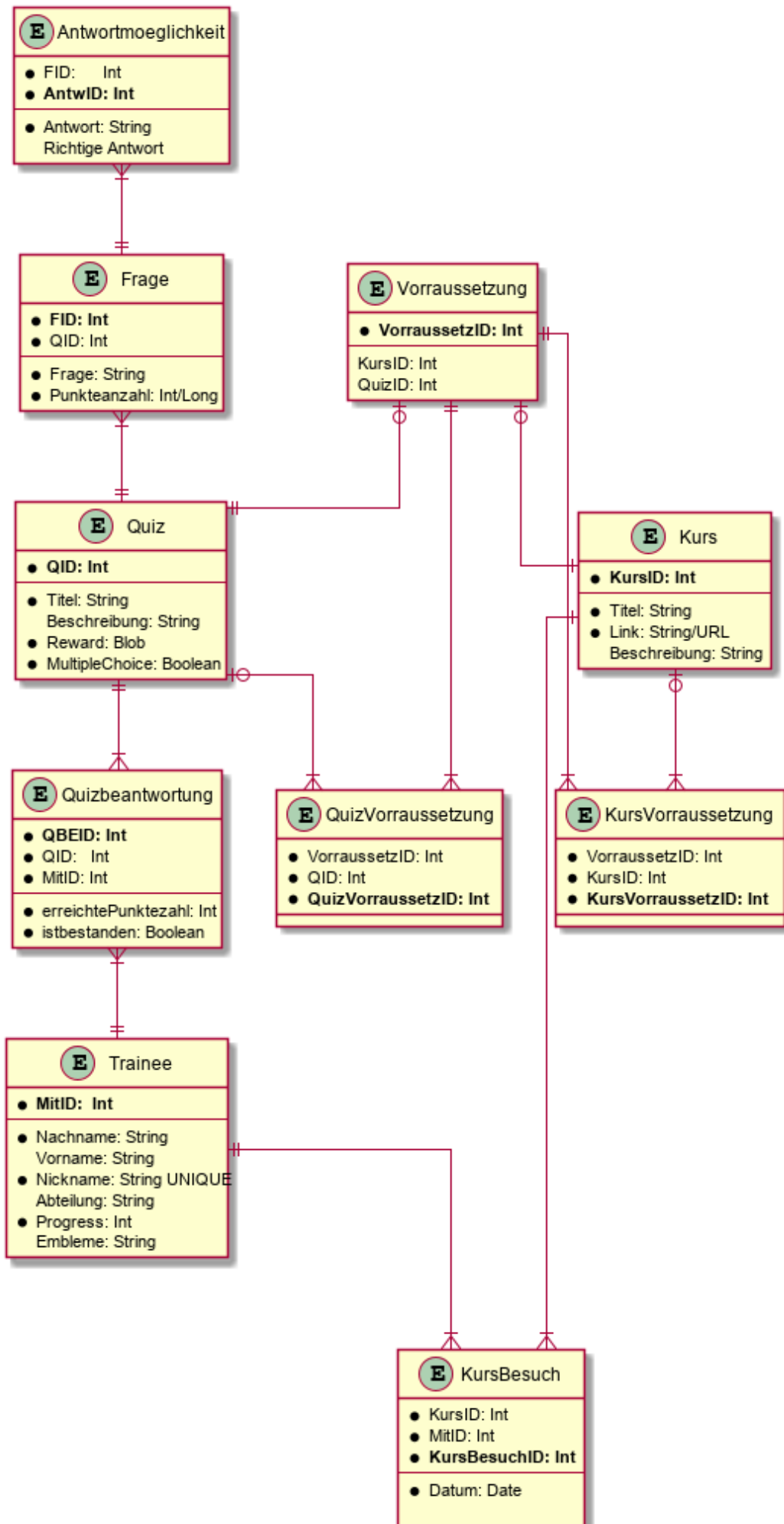
All this vocabulary is completely unknown in an object-oriented language such as Java. In Java, we manipulate objects that are instances of classes. Objects inherit from others, have references to collections of other objects, and sometimes point to themselves in a recursive manner. We have concrete classes, abstract classes, interfaces, enumerations, annotations, methods, attributes, and so on. Objects encapsulate state and behavior in a nice way, but this state is only accessible when the Java Virtual Machine (JVM) is running: if the JVM stops or the garbage collector cleans its memory content, objects disappear, as well as their state. Some objects need to be persistent. By persistent data, I mean data that are deliberately stored in a permanent form on magnetic media, flash memory, and so forth. An object that can store its state to get reused later is said to be persistent. The principle of object-relational mapping (ORM) is to bring the world of database and objects together. It involves delegating access to relational databases to external tools or frameworks, which in turn give an objectoriented view of relational data, and vice versa. Mapping tools have a bidirectional correspondence between the database and objects. Java Persistence API (JPA) is the preferred technology and is part of Java EE 7.

(vgl. *Beginning Java EE-Java Persistence API*, Antonio Goncalves, 2013)

## 13.3. Getting Started

The first task was to visually showcase how all the tables needed, connect with each other, so we can understand what we need to do, and how everything should look and interact. Here we used the diagram called ER-Modell which does exactly that for us.

### 13.3.1. ER-Modell





The ER-Model in general is a diagram, which shows the relationships between the entities of a database.

In this chapter I will showcase all the tables with their attributes and their respective meanings. Starting from the bottom to the top.

## **Trainee**

The first table is the table "Trainee". This table contains all trainees. It contains their first name, last name, the nickname they use on the website, their department in the company and their progress they have by doing quizzes. Also every Trainee can get emblems by completing quizzes. It has a One-To-Many relationship to the table "Quizbeantwortung" and to the table "KursBesuch".

## **KursBesuch**

The entity "KursBesuch" is a table, which splits the many-to-many relationship between "trainee" and "kurs" into two separate One-To-Many relationships therefore its relationships are Many-To-One to "Trainee" and to "Kurs". That means it has the PK (primary key) of both tables as a foreign key. It also provides a column named "datum" which indicates the date the user has visited the course. The function of this entity is to show which trainee has done which course and when.

## **KursVoraussetzung**

The function of "KursVoraussetzung" is that a course can have a course or a quiz as a requirement to take this course. This table contains courses and their required quizzes or courses. Therefore it contains the "KursVoraussetzID", the "KursID" and the "VoraussetzID" as columns.

## **QuizVoraussetzung**

This table has the same function as "KursVoraussetzung", but instead of courses with quizzes. So this table contains quizzes and the requirements to take this quiz, which can be courses or previous quizzes.

## **Quizbeantwortung**

This entity is essential for the implementation of the "Voraussetzungs-Logik". We need this table to check if a trainee has already done a quiz. This is done with the column "istbestanden". Also it stores the points a trainee has achieved in the quiz. It also splits up a Many-To-Many relationship between "Quiz" and "Trainee" into two One-To-Many relationships.

## **Kurs**

Here is the Kurs-Entity, it has the primary key "KursID", which helps to find a specific course from the table. The table also has a column "titel" which is basically the title of this course. It also has the constraint NOT NULL so if there is a new course there must be a title as well. The next column is the "link" it contains the links of the courses so the visitors can be redirected to the page with the explanation of the topic. This cannot be empty too. The last column is "beschreibung", which is the description of the course. This column can be empty, but it is not recommended since it helps the user specify which course contains what information.

## Quiz

This is the table for all quizzes. It contains the "QID", which is the primary key, a title, a description and a reward. The title has a constraint called NOT NULL because every quiz must have a specification which topic it has. There is also a option if the quiz is multiple choice. There are also so called queries to find data with specific parameters from the database. These queries are defined in the associated entity class.

## Voraussetzung

This table defines the requirement a course or a quiz can have. Therefor it contains a "KursId" and a "qid". The primary key of this table is then given to "KursVoraussetzung" or "QuizVoraussetzung" respectively.

## Frage

Every Quiz contains many questions this means that "Frage" and "Quiz" have a Many-To-One relationship. The questions are stored in the table "Frage". Each question has a "FID", a "QID", so it can be connected to a quiz. A question also has the questiontext itself and points you get for each question.

## Antwortmoeglichkeiten

Every question has 4 answers this means that there is a Many-To-One relationship between "Antwortmoeglichkeiten" and "Frage". These answers are stored in this table. Each answer is connected to its question. Also the answertext itself is stored here, as well as a boolean value if the answer is correct or not.

## 13.3.2. EJBs

After knowing how to connect the tables with each other, the next step was to create so called EJBs for every table we needed one for.

EJBs are components that summarize the business logic and take care of transactions and security. It is basically a connection to the database.

They must contain an annotation called "Stateless", this means that after a methode is called it is terminated. This is the usual way to annotate a EJB.

Every EJB has a methode to find exactly one entity, persist/merge an entity or delete one entitiy from the database.

*KursEJB.java*

```
public List<Kurs> findAll() {  
    return em.createNamedQuery(Kurs.QUERY_FINDALLKURSE, Kurs.class)  
        .getResultList();  
}
```

The main use of this EJB is to provide the courses page with all the courses within the database.

This is done with the "findAll" methode. It is used on the courses page where all courses available are shown.

*FrageEJB.java*

```
@Stateless
public class FrageEJB {
    @PersistenceContext(unitName = "Diplomarbeit")
    private EntityManager em;
    private Frage frage;

    public Frage find(String FID) {
        return em.find(Frage.class, FID);
    }

    public List<Frage> findAll() {
        return em.createNamedQuery(Frage.QUERY_FINDALLFRAGEN, Frage.class)
            .getResultList();
    }

    public void update(Frage f) {
        em.merge(f);
    }

    public void delete(int FID) {
        em.getTransaction().begin();
        Frage f = em.getReference(Frage.class, FID);
        em.remove(f);
        em.getTransaction().commit();
    }

    public List<Frage> findFrageByQID(String qid) {
        return em.createNamedQuery(Frage.QUERY_FINDFRAGENZUQID, Frage.class)
            .setParameter("QID", qid).getResultList();
    }
}
```

The use of the "FrageEJB" is for example to find all questions connected to a qid. This is used to show the questions when you take a quiz. Also it is available to only find one question or all questions in form of a list.

```
@Stateless
public class QuizEJB {
    @PersistenceContext(unitName = "Diplomarbeit")
    private EntityManager em;

    public Quiz find(String QID) {
        return em.find(Quiz.class, QID);
    }

    public List<Quiz> findAll() {
        return em.createNamedQuery(Quiz.QUERY_FINDALL, Quiz.class).getResultList();
    }

    public void update(Quiz q) {
        em.merge(q);
    }

    public void delete(int QID) {
        em.getTransaction().begin();
        Quiz q = em.getReference(Quiz.class, QID);
        em.remove(q);
        em.getTransaction().commit();
    }
}
```

The function of this EJB is to find all quizzes from the database and return them in a list to the website. Also you can find single quizzes with the methode "find", you can delete and update quizzes.

```
@Stateless
public class TraineeEJB {
    @PersistenceContext(unitName = "Diplomarbeit")
    private EntityManager em;

    public Trainee find(String MitID) {
        return em.find(Trainee.class, MitID);
    }

    public List<Trainee> findAll() {
        return em.createNamedQuery(Trainee.QUERY_FINDALLTRAINEES, Trainee.class)
            .getResultList();
    }

    public void update(Trainee t) {
        em.merge(t);
    }

    public void delete(int MitID) {
        em.getTransaction().begin();
        Trainee t = em.getReference(Trainee.class, MitID);
        em.remove(t);
        em.getTransaction().commit();
    }
}
```

The use of this class is to provide all the trainees deposited on the database to the associated web page. As in every other EJB there is an opportunity to find, update or delete a trainee.

```
@Stateless
public class AntwortmoeglichkeitenEJB {
    @PersistenceContext(unitName = "Diplomarbeit")
    private EntityManager em;

    public Antwortmoeglichkeiten find(int AntwID) {
        return em.find(Antwortmoeglichkeiten.class, AntwID);
    }

    public void update(Antwortmoeglichkeiten antw) {
        em.merge(antw);
    }

    public void delete(int AntwID) {
        em.getTransaction().begin();
        Antwortmoeglichkeiten antw = em.getReference(Antwortmoeglichkeiten.class,
        AntwID);
        em.remove(antw);
        em.getTransaction().commit();
    }

    public List<Antwortmoeglichkeiten> findAntwortenByFID(String fid) {
        return em.createNamedQuery(Antwortmoeglichkeiten.QUERY_FINDANTWORTEN_BYFID,
        Antwortmoeglichkeiten.class)
            .setParameter("FID", fid).getResultList();
    }
}
```

This EJB provides the page where you take the quiz with all the corresponding answers to the questions. This is done by the methode "findAntwortenByFID" which starts a query when invoked. This query is defined in the "Antwortmoeglichkeiten" entity.

```
@Stateless
public class QuizVoraussetzungEJB {
    @PersistenceContext(unitName = "Diplomarbeit")
    private EntityManager em;

    public QuizVoraussetzung find(String QuizVoraussetzID) {
        return em.find(QuizVoraussetzung.class, QuizVoraussetzID);
    }

    public void update(QuizVoraussetzung QuizVoraussetzID) {
        em.merge(QuizVoraussetzID);
    }

    public void delete(String QuizVoraussetzID) {
        em.getTransaction().begin();
        QuizVoraussetzung quizvoraussetzung = em.getReference(QuizVoraussetzung.class,
QuizVoraussetzID);
        em.remove(quizvoraussetzung);
        em.getTransaction().commit();
    }

    public List<QuizVoraussetzung> findAllQuizVoraussetzungen(String qid) {
        return em.createNamedQuery(QuizVoraussetzung.QUERY_FINDALLVORAUSSETZUNGEN,
QuizVoraussetzung.class)
            .setParameter("QID", qid).getResultList();
    }
}
```

The "QuizVoraussetzungsEJB" allocates all "QuizVoraussetzungen" with the "findAllQuizVoraussetzungen" methode. It is used in the "Voraussetzungs-Logic".

```
@Stateless
public class VoraussetzungEJB {
    @PersistenceContext(unitName = "Diplomarbeit")
    private EntityManager em;

    public Voraussetzung find(String VoraussetzID) {
        return em.find(Voraussetzung.class, VoraussetzID);
    }

    public void update(Voraussetzung v) {
        em.merge(v);
    }

    public void delete(int VoraussetzID) {
        em.getTransaction().begin();
        Voraussetzung v = em.getReference(Voraussetzung.class, VoraussetzID);
        em.remove(v);
        em.getTransaction().commit();
    }
}
```

This EJB is used for the requirements logic. It contains a find method where you can search for a requirement via a "VoraussetzID", which then returns a "Voraussetzung"-Entity.



```
@Stateless
public class QuizbeantwortungEJB {
    @PersistenceContext(unitName = "Diplomarbeit")
    private EntityManager em;

    public Quizbeantwortung find(String qbeid) {
        return em.find(Quizbeantwortung.class, qbeid);
    }

    public void update(Quizbeantwortung quizbeantw) {
        em.merge(quizbeantw);
    }

    public void delete(String qbeid) {
        em.getTransaction().begin();
        Quizbeantwortung quizbeantw = em.getReference(Quizbeantwortung.class, qbeid);
        em.remove(quizbeantw);
        em.getTransaction().commit();
    }

    public List<Quizbeantwortung> findByQIDAndMITID(String qid, String mitid) {
        return em.createNamedQuery(Quizbeantwortung.QUERY_FINDBY_QIDANDMITID,
            Quizbeantwortung.class)
            .setParameter("QID", qid).setParameter("MitID", mitid).getResultList(
        );
    }
}
```

The use of this bean is to provide information to all the quizzes a specific trainee has taken and when. This is done by a query, which is defined in the entity of this bean. It is required in the requirement logic.

### 13.3.3. Persistence.xml

*persistence.xml*

The "persistence.xml" is used to configure many things, such as the source of the script used for dropping, if the database should always drop and then create all tables, and much more.

### 13.3.4. SQL-Files

In this chapter I will introduce all SQL-files used in this project in the order they are runned when the program is started.

#### Dropping all tables

This script has the function of dropping every table before creating so there are no errors.

## **Creating the tables**

This file is used to create all the tables we are using. It produces every table with its associated columns. This file is also used to give all the tables its associated primary keys and foreign keys.

## **Insert of data**

Last there is the "insertSQL" script, which is used to insert all the data we want into the belonging table. The order of the commands is very important because, if you insert "antwortmoeglichkeiten" at first there would be no matching "FNR" or with "frage " no fitting "QID".

# 14. Bibliography

- Bandura, A.** (1986). Social Foundations of Thought and Action. Englewood Cliffs, NJ: Prentice-Hall.
- Büser, T.** (2003). Offene Angebote an geschlossene Systeme - Überlegungen zur Gestaltung von Lernumgebungen für selbstorganisiertes Lernen aus Sicht des Konstruktivismus. In U. Witthaus, W. Wittwer & C. Espe, Selbst gesteuertes Lernen (S. 27-41). Bielefeld: wbv.
- Creß, U.** (?). Lernorientierungen, Lernstile, Lerntypen und kognitive Stile. In "Handbuch Lernstrategien" von Heinz Mandl & Helmut Felix Friedrich.
- Dunn, R. & Price, G.** (1989). Learning Styles Inventory. Lawrence, KS: Price Systems.
- Goncalves A.** (2013). Beginning Java EE 7. Apress.
- Heitmann, K.** (2012). Wissensmanagement in der Schulentwicklung Theoretische Analse und empirische Exploration aus systemischer Sicht, Springer VS.
- Hochleitner, P.** (2016). Erforschung der Lerntypen und -strategien am Beispiel einer Handelsschulklasse. Akademiker Verlag.
- Huang et al** (2013). A Practitioner's Guide To Gamification Of Education. Rotman School of Management, University of Toronto.
- Kolb, D.** (1984), Experience as the source of leaning and development. Englewood Cliffs: Prentice-Hall.
- Niegemann, H. M. & Hofer, M.** (1997). Ein Modell selbstkontrollierten Lernens und über die Schwierigkeiten, selbstkontrolliertes Lernen hervorzubringen. In H. Gruber, A. Renkl, Wege zum Können. Determinanten des Kompetenzerwerbs (S. 263-280). Göttingen: Huber.
- Pask, G. & Scott, B.** (1972). Learning strategies and individual competence. International Journal of Man-Machine Studies, 4, 217-253.
- Rosendahl, J.** (2010). Selbstreguliertes Lernen in der dualen Ausbildung. W. Bertelsmann Verlag GmbH & Co. KG, Bielefeld, 2010.
- Sary, C. & Maroscher, M. & Sary, E.** (2013). Wissensmanagement in der Praxis - Methoden Werkzeuge Beispiele, Hanser.
- Vester F.** (1998). Denken, Lernen, Vergessen (25. Auflage). München: dtv.
- Weinstein, C. E. & Mayer, R. E.** (1986). The Teaching of Learning Strategies. In M. C. Wittrock, Handbook of Research on Teaching (S. 315-327). London: Collier Macmillan Publishers.

# 15. List of figures

**Fig. 1:** Learning styles according to Kolb

**Fig. 2:** Sample Repertory Grid

**Fig. 3:** System Context diagram for the learning platform "GAME"

**Fig. 4:** System Container diagram for the learning platform "GAME"

**Fig. 5:** User Interface concept

**Fig. 6:** JSF architecture diagram

**Fig. 7:** Design of the GAME platform

**Fig. 8:** Design of the courses page

**Fig. 9:** Design of the quizzes page

**Fig. 10:** Quizzes page with fulfilled requirements

**Fig. 11:** Design of the page, where the user takes a single-choice quiz

**Fig. 12:** Bottom half of the single-choice quiz

**Fig. 13:** Design of the page, where the user takes a multiple-choice quiz

**Fig. 14:** Design of the page, where the user sees his/her quiz-results

**Fig. 15:** Trainers page

**Fig. 16:** Emblemtafel page

**Fig. 17:** ER-Model

# 16. Timetables

## 16.1. Eric Haneder

Date	Start time	Activity	Duration
24.06.2019	15:00	Diplomarbeitsbesprechung	2 h
04.07.2019	14:00	Diplomarbeitsbesprechung	3 h
06.07.2019	15:00	Introduction, Stakeholders documented	4 h
10.07.2019	16:00	first chapters completed	3 h
18.07.2019	10:00	Diplomarbeitsbesprechung	2 h
20.07.2019	15:00	Documentation optimated to clients interest	3 h
30.07.2019	10:00	Diplomarbeitsbesprechung	2 h
02.09.2019	14:00	started market research	5 h
20.09.2019	16:00	continued market research	4 h
26.09.2019	15:00	Diplomarbeitsbesprechung	2 h
28.09.2019	15:30	System Scope and Context, ADRs	5 h
02.10.2019	15:00	C4-model created	4 h
03.10.2019	22:00	completion of ongoing tasks	1 h
10.10.2019	15:00	Diplomarbeitsbesprechung	2,5 h
10.10.2019	18:30	discussion of the next steps	1 h
22.10.2019	20:00	Design Decisions created	2 h
02.11.2019	19:00	creating first design ideas for the site	3 h
10.11.2019	19:00	completing the design diagrams	4 h

21.11.2019	15:00	Diplomarbeitsbesprechung	2 h
23.11.2019	18:00	realizing the design/structure	3 h
01.12.2019	16:00	realizing the design/structure	6 h
02.12.2019	17:00	creation of quiz classes/logic	1,5 h
05.12.2019	15:00	Diplomarbeitsbesprechung	2,5 h
16.12.2019	18:00	fixing bugs on the pages	1 h
18.12.2019	19:30	first testquiz classes created and tested	3,5 h
19.12.2019	19:00	Reading Java EE book	1 h
21.12.2019	18:00	created Trainee classes, added new logic to takequiz	7 h
22.12.2019	19:00	Reading Java EE book	2 h
23.12.2019	18:00	Reading Java EE book	1,5 h
26.12.2019	19:30	Reading Java EE book	1 h
29.12.2019	19:00	Reading Java EE book	2 h
01.01.2020	16:00	Reading Java EE book	2 h
02.01.2020	19:00	Theorie writing: Lerntypen	3 h
03.01.2020	17:00	Theorie/Praxis writing	4 h
07.01.2020	16:00	new head & footer for the Uis, emblemboard created	3 h
08.01.2020	18:00	Praxis written, emblemboard developed	3 h
09.01.2020	18:00	added a testpackage "testquiz" to test some quiz logic	4 h
10.01.2020	20:00	Theorie writing	2 h
11.01.2020	20:00	design developed, started the takequiz page	3 h
12.01.2020	14:00	developed the takequiz page	3,5 h
13.01.2020	18:00	bugfixes on result.xhtml	2 h

14.01.2020	18:00	Documentation written, QuizController developed	3,5 h
15.01.2020	18:00	Documentation written	3 h
16.01.2020	19:00	Documentation written	1 h
21.01.2020	16:00	Design further developed, changed quiz appearance	4 h
22.01.2020	15:00	Diplomarbeitsbesprechung	4 h
23.01.2020	15:00	Mainlyout further designed (header, footer)	3 h
26.01.2020	19:00	Documentation written	4 h
27.01.2020	18:00	updated leaderboard, added multipleChoice page	4 h
28.01.2020	16:00	Theorie weitergeschrieben	2,5 h
29.01.2020	19:00	Theorie weitergeschrieben	1 h
11.02.2020	15:00	12. Diplomarbeitsbesprechung	3 h
13.02.2020	10:00	creating and formating emblems	1 h
16.02.2020	15:00	splitted courses & quizzes, QuizController optimated	3 h
18.02.2020	16:00	added f:param handover to takequiz via Backing Bean	2 h
19.02.2020	16:00	updated result.xhtml, added KursController	3 h
20.02.2020	15:00	Diplomarbeitsbesprechung	3 h
21.02.2020	18:00	Documentation updated (new logo, abstract)	2 h
22.02.2020	17:00	bugfixing in documentation	1 h
23.02.2020	18:00	Documentation written	2 h
24.02.2020	19:00	Documentation written	1 h

25.02.2020	19:00	Database connection set up	1 h
28.02.2020	13:00	Diplomarbeitsbesprechung	5 h
01.03.2020	14:00	database connection implemented,	4 h
02.03.2020	18:00	added a description on every page	1 h
05.03.2020	15:00	documentation written	1 h
06.03.2020	20:00	QuizController class modified (added trainee)	1 h
07.03.2020	14:00	bugfixes in Quizcontroller, deleted deprecated code	2 h
08.03.2020	18:00	documentation written	1 h
11.03.2020	16:00	fixed menubar, multiplechoice quiz page updated	3 h
14.03.2020	13:00	Documentation written	3 h
15.03.2020	10:00	Documentation finished	5 h
TOTAL			194 h



## 16.2. Jan Binder

Datum	Tätigkeit	Stunden
15.06.2019	1ste Besprechung	2,00
04.07.2019	Besprechung	3,00
	Ausarbeitung der in der Besprechung besprochenen Aufgaben	6,00
18.07.2019	Besprechung	2,00
	Ausarbeitung der in der Besprechung besprochenen Aufgaben	7,00
30.07.2019	Besprechung	2,00
	Ausarbeitung der in der Besprechung besprochenen Aufgaben	7,00
02.10.2019	C4 Modell fertigstellen	4,00
23.10.2019	Kontrollieren auf korrekte Installation aller benötigten Programme	1,00
24.10.2019	Besprechung	2,50
30.10.2019	Erste Besprechung bezüglich Websiteerstellung	2,00
02.11.2019	Website Design Prototyp erstellen (mit Hilfe von Wix)	3,00
10.11.2019	Website Design und erste Versuche dies zu programmieren	5,00
21.11.2019	Besprechung	3,50
04.12.2019	ER-Modell erstellen	3,00
05.12.2019	Besprechung	2,50
17.12.2019	ER-Modell normalisieren	3,00
18.12.2019	Weiterarbeiten an der Darstellung des ER-Modells	3,00
19.12.2019	Besprechung	3,00
20.12.2019	ER-Modell in AsciiDoc dargestellt	2,00
21.12.2019	Gemeinsames Ausarbeiten der Aufgaben	7,00
27.12.2019	Durchsuchen passender Kurse für JPA auf Udemy	2,00

28.12.2019	Anschauen der ersten Lektionen des Kurses über JPA inkl Installation der benötigten Software für die Durchführung der Praxisbeispiele	2,00
02.01.2020	Weiterschauen der Lektionen des Kurses über JPA	2,00
06.01.2020	Weiterschauen der Lektionen des Kurses über JPA	2,00
13.01.2020	Kurse und Fragen ausarbeiten; ersten Entities erstellen	2,00
14.01.2020	Kurse und Fragen ausarbeiten	2,00
15.01.2020	Kurse und Fragen ausarbeiten	2,00
19.01.2020	Weiterschauen der Lektionen des Kurses über JPA; Kurse und Fragen ausarbeiten	2,00
22.01.2020	Besprechung+Planung	4,00
23.01.2020	Entities erstellt	3,00
25.01.2020	Weiterschauen der Lektionen des Kurses über JPA	1,50
26.01.2020	Weiterschauen der Lektionen des Kurses über JPA+Gemeinsames Ausarbeiten der geplanten Aufgaben	5,50
02.02.2020	Erste Ansätze in Bezug auf EJBs	2,00
03.02.2020	EJBs erweitert	2,00
07.02.2020	Gemeinsames Ausarbeiten der Aufgaben	2,00
11.02.2020	Besprechung	3,00
15.02.2020	Tabellen in Datenbank erstellt und Testdatensätze eingefügt	3,00
16.02.2020	Gemeinsames Ausarbeiten der Aufgaben	3,00
22.02.2020	Dokumentation schreiben, Fehlerbehebung bei den SQL Dateien	8,00
23.02.2020	Dokumentation schreiben, Fehlerbehebung bei den SQL Dateien	3,50
25.02.2020	Dokumentation schreiben	2,50
26.02.2020	Datenbankanbindungsfehler versuchen zu beheben	2,00

27.02.2020	Besprechung und anschließend Datenbankbindung verbessern	6,00
29.02.2020	Theoretische Dokumentation updated	2,00
01.03.2020	Fehler in der Datenbankbindung behoben, weiters Datenbankbindung verbessern	4,00
02.03.2020	ModellCreators modified	2,00
05.03.2020	Besprechung	4,00
06.03.2020	Voraussetzung, Frage, Quiz, Quizbeantwortung, TestQuizController, Traineecontroller modified	2,50
07.03.2020	TraineeController error and Voraussetzungslogik fixed	5,00
08.03.2020	Dokumentation weitergeschrieben	3,00
09.03.2020	Dokumentation weitergeschrieben	3,00
11.03.2020	Multiple Choice implementiert, Quiz und Fragen in database written	3,00
12.03.2020	Besprechung	3,00
14.03.2020	Dokumentation updaten	4,00
15.03.2020	Dokumenation updaten	4,00