Table of Contents

Praktische Aufgabenstellung: Jan Binder 1
Back End: Datenbank und Datenbankverbindung 1
Begin 1
Grundlagen 1
Getting Started 2

Praktische Aufgabenstellung: Jan Binder

Back End: Datenbank und Datenbankverbindung

Begin

Zu aller erst wurden die allgemeinen Grundlagen, wie die Technologie, die verwendet wird, und das Design festgelegt. Auch die Datenbank, in der die entsprechenden Tabellen und Datensätze stehen soll, wurde vereinbart.

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.

Grundlagen

Man braucht, um eine Datenbank und eine Datenbankverbindung zu erzeugen, ein gewisses Vorwissen. Hier werden SQL, JPA und Java von Nöten sein. Also wie das Programm mit der Datenbank kommuniziert und wie sogenannte Queries erstellt werden.

You need some foreknowledgement to build a database and a database connectivity. Knowlegde about SQL, JPA and Java is required. You need these languages to create queriesor to understand how the program communicates with the database.

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.

JPA

Getting Started

Entities

Kurs Entities

Kurs.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/kurs/Kurs.java[lines=19..40]
//Getter, Setter
```

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.

KursBesuch.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/kurs/KursBesuch.java[lines=22..44]

//Getter, Setter
```

The entity "KursBesuch" is a table, which splits the many-to-many relationship between "trainee" and "kurs". That means it has the PK (primary key) of both tables as a foreign key and 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.

Vorausetzung Entities

Vorausetzung.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/kurs/Voraussetzung.java[lines=19..37]

//Getter, Setter
```

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

KursVorausetzung.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/kurs/KursVoraussetzung.java[lines=19..36]
//Getter, Setter
```

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.

QuizVorausetzung.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/QuizVoraussetzung.java[lines=21..36]

//Getter, Setter
```

This table has the same function as "KursVoraussetzung", but instead of courses with quizzes. So this table contains quizzes and their required courses or quizzes.

Quiz Entities

Quiz.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/Quiz.java[lines=25..53]

//Getter, Setter
```

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 are also so called queries to find data with specific parameters from the database.

Quizbeantwortung.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/Quizbeantwortung.java[lines=22..45]


//Getter, Setter
```

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".

Frage.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/Frage.java[lines=21..52]

//Getter, Setter
```

Every Quiz contains of many questions. These 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 question itself and points you get for each question.

Antwortmoeglichkeiten.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/Antwortmoeglichkeiten.java[lines=20..45]

//Getter, Setter
```

Every question has 4 answers. 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.

Trainee Entity

Trainee.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/trainee/Trainee.java[lines=25..53]

//Getter, Setter
```

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.

EJBs

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

KursEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/kurs/KursEJB.java[lines=16..]
```

The main use of the EJB is to provide the courses page with all the courses within the database. This is done with the "findAll" methode.

FrageEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/FrageEJB.java[lines=21..]
```

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.

QuizEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/QuizEJB.java[lines=16..]
```

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.

TraineeEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/trainee/TraineeEJB.java[lines=16..]
```

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.

AntwortmoeglichkeitenEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/AntwortmoeglichkeitenEJB.java[lines=16..]
```

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.

QuizVoraussetzungEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/QuizVoraussetzungEJB.java[lines=13..]
```

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

VoraussetzungEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/kurs/VoraussetzungEJB.java[lines=16..]
```

This EJB is used for the requirements logic. It contains a find methode where you can search for a requirement via a "VoraussetzID", which then returns a "Voraussetzung"-Entity. Also it has a update and delete methode.

QuizbeantwortungEJB.java

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/java/org/game/trainee/quiz/QuizbeantwortungEJB.java[lines=13..]
```

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 and it is required in the requirement logic.

Persistence.xml

persistence.xml

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-
web/src/main/resources/META-INF/persistence.xml[]
```

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

SQL-Files

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

dropSQL.sql

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/resources/META-INF/dropSQL.sql[]
```

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

createSQL.sql

```
Unresolved directive in Praxis_Jan.adoc - include::../../game-trainee-web/src/main
/resources/META-INF/createSQL.sql[]
```

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.

insertSQL.sql

```
INSERT INTO game.trainee ("mitid", "nachname", "vorname", "nickname", "abteilung",
"progress", "embleme") VALUES ('1', 'Binder', 'Jan', 'Syreax', 'Projektmanagement',
500, NULL);

INSERT INTO game.quiz ("qid", "titel", "beschreibung", "reward", "multiplechoice")
VALUES ('1', 'Start', 'Quiz about the general knowledge of Java', 'test', false);

INSERT INTO game.frage ("fid", "qid", "frage", "punktezahl") VALUES ('1', '1', 'What
are advantages of Java?', 10);

INSERT INTO game.antwortmoeglichkeiten ("antwid", "fid", "antwort", "richtigeantwort")
VALUES ('1', '1', 'Flawless', false);

INSERT INTO game.quizbeantwortung ("qbeid", "qid", "mitid", "erreichtepunktezahl",
"istbestanden") VALUES ('1', '1', '1', 0, false);

INSERT INTO game.kurs ("kursid", "titel", "link", "beschreibung") VALUES ('1', 'Start-
Kurs', 'https://www.tutorialspoint.com/java/index.htm', 'Dieser Kurs ist der erste
Kurs der abgelegt werden muss. Er erklärt dir die Basics von Java.');

INSERT INTO game.voraussetzung("voraussetzid", "kursid", "qid") VALUES ('1', null, '1
');

INSERT INTO game.quizvoraussetzung("quizvoraussetzid", "voraussetzid", "qid") VALUES
('1', '1', '2');
```

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 existing matching "FNR" or with "frage " no fitting "QID". Here are only the first insert of every table because otherwise there would be too much code, which would make it too complex.