Dokumentation

Teammitglied: Daniel Rohrwild

Rolle:

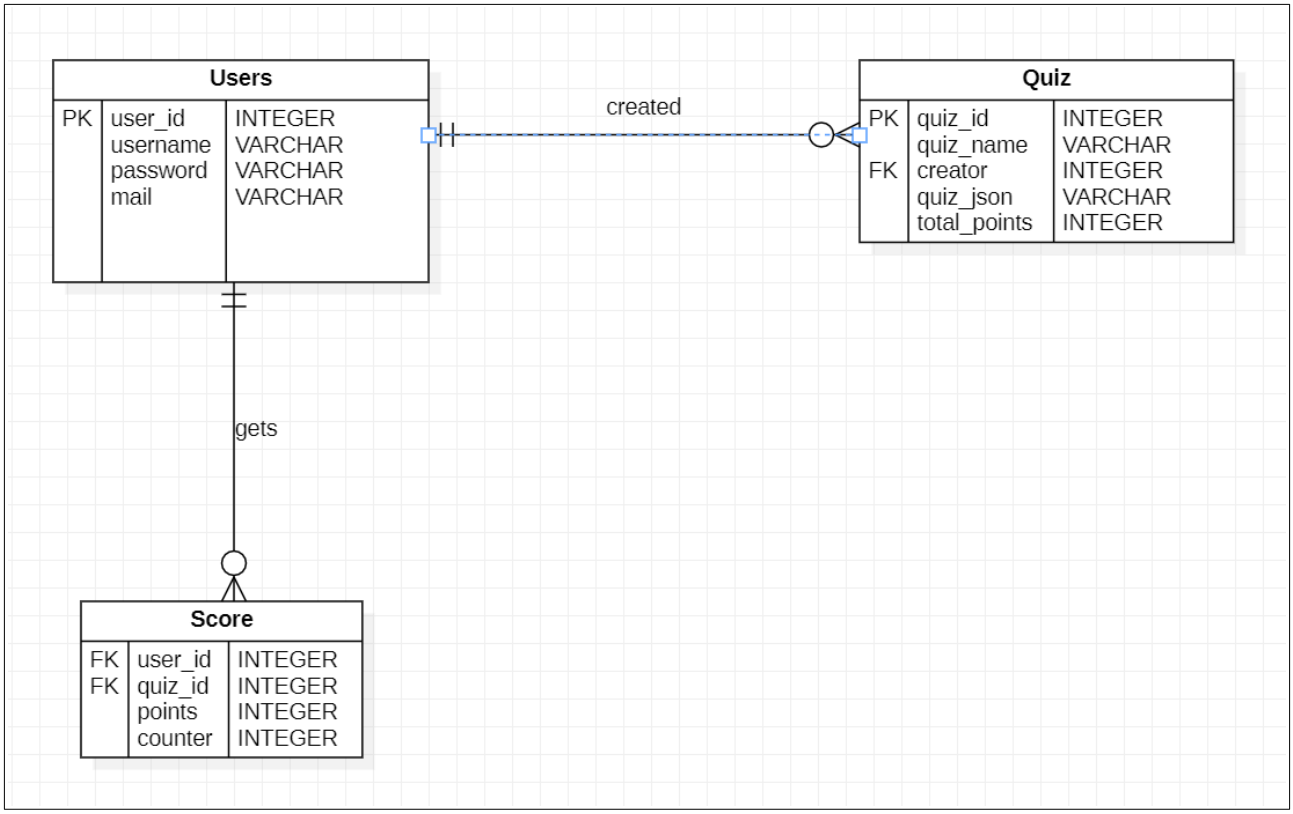
- Datenbank-Verantwortlicher

Aufgaben Übersicht:

1. ER-Diagramm erstellen
2. DB-Skript erstellen
3. Code erstellen:
   3a. play.php
   3b. check.php
   3c. rangliste.php

1. ER-Diagramm erstellen

Um eine grobe Übersicht über den Aufbau und die Beziehungen der Datenbank zu bekommen wurde zu Beginn des Projekts ein ER-Diagramm erstellt, welches nachfolgend abgebildet ist:



Für das Projekt wurden 3 Tabellen angelegt.
Die Tabelle Users hat als Primärschlüssel die Spalte user_id (INTEGER). Die user_id hat in der Datenbank die Eigenschaft auto_increment, d.h. die ID wird bei jedem neuen Eintrag um eins erhöht, sodass jeder User eine andere ID erhält. Neben der user_id werden vom User noch der vom User vergebene Username (VARCHAR), das Passwort(VARCHAR) und die Email(VARCHAR) gespeichert, die bei der Registrierung in die Datenbank gespeichert werden.

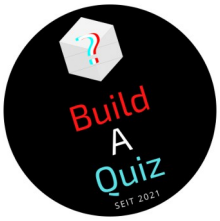
In der Tabelle Quiz sind alle Quizze gespeichert. Als Primärschlüssel wurde die Spalte quiz_id (INTEGER) erstellt, die genau wie die user_id auch die Eigenschaft auto_increment hat, sodass jedes Quiz eine eindeutige Nummer hat.

Es werden zudem der quiz-name (VARCHAR) gespeichert, der vom Quizersteller frei gewählt werden kann. Der Quiz-Ersteller wird in der Spalte creator(INTEGER) als Fremdschlüssel auf die user_id der Tabelle Users referenziert. Das erstellte Quiz wird im JSON-Format in der Spalte quiz_json (VARCHAR) gespeichert. In der Spalte total_points (INTEGER) wird die Gesamtpunktzahl des Quizes gespeichert.

Die Relation zwischen Users und Quiz ist wie folgt: 1:N. Ein User kann mehrere Quizze erstellen, aber 1 Quiz wird von nur einem User erstellt.

Als dritte und letzte Tabelle wurde die Tabelle Score angelegt, welche einen zusammengesetzten Primärschlüssel hat, der aus der user_id (INTEGER) der Tabelle Users und der quiz_id (INTEGER) der Tabelle Quiz besteht. In dieser Tabelle findet die Zuordnung von Usern und Quiz statt. Wenn ein User vom Creator eines Quizzes eingeladen wird, das Quiz spielen zu dürfen, wird in der Tabelle Score ein Eintrag gespeichert. Dieser Eintrag stellt die Freigabe des Quizzes dar. Neben den beiden Fremdschlüssel-Spalten user_id und quiz_id existieren noch die Spalten points (INTEGER) und counter (INTEGER). Die Spalte points gibt die erreichte Punktzahl an, die ein User beim Spielen eines Quizzes erreicht hat. Der Counter wurde erstellt um zu prüfen, ob ein User zu einem Quiz eingeladen wurde und das Quiz gespielt hat oder noch nicht. Wenn das Quiz das erste mal gespielt wird, wird der Counter auf 1 gesetzt. Diese Spalte ist wichtig, denn der User wird auf der Ranglistenseite erste angezeigt, wenn er das Spiel das erste mal gespielt hat. Hat er das Quiz noch nicht gespielt, wird er auch in der Rangliste noch nicht angezeigt.

Die Beziehung zwischen den Tabellen Users und Score ist eine 1:N-Beziehung. Ein User kann mehrere Scores haben, da er mehrere Quizze spielen kann, für jedes Quiz hat er aber nur einen Score.



2. DB-Skript erstellen

Um sicher zu gehen, dass jedes Teammitglied die selbe Datenbank zum Testen und Entwickeln hat wurde ein Datenbankskript erstellt, welches mit Copy&Paste, die Datenbank, Tabellen, User und Spalten mit den entsprechenden Primär- und Fremdschlüsseln. Zudem wurden einige Testdaten z.B. Testuser angelegt. Zudem wurden einige Löschbefehle dazugeschrieben, die bei Bedarf kopiert und als SQL Befehl an die DB gesendet werden können.

DB-Skript:

```
#Skript für DB-Erstellung
#Version 3.0

#DB anlegen
create database baq;

#alle Datenbanken anzeigen:
#show databases;

#User 'baq' anlegen mit PW 'baq123'
create user 'baq'@'localhost' identified by 'baq123';

#alle User anzeigen:
#select User from mysql.user;

#User 'baq' der DB zuweisen und alle Rechte geben:
grant all on baq.* to baq@localhost identified by 'baq123';

#Berechtigung anzeigen:
#show grants for 'baq'@'localhost';
```

```
#Tabelle users anlegen:
create table users ( user_id int primary key AUTO_INCREMENT,
                username varchar(15),
                password varchar(15),
                mail varchar(30));

#Tabelle quiz anlegen:
create table quiz ( quiz_id int primary key AUTO_INCREMENT,
                quiz_name varchar(20),
                creator integer not null REFERENCES users(user_id),
                quiz_json varchar(5000),
                total_points integer);

#Tabelle score anlegen:
create table score( user_id int not null REFERENCES users(user_id),
                   quiz_id int not null REFERENCES quiz (quiz_id),
               points int not null,
               counter int);

#Testuser anlegen:
insert into users (username, password, mail) values ('luke', 'luke',
'luke@baq.de');
insert into users (username, password, mail) values ('lea', 'lea',
'lea@baq.de');

#DB löschen:
#drop database 'baq';

#User löschen:
# DROP USER 'baq'@'localhost';

#nur Berechtigungen löschen:
#REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'baq'@'localhost';
```

3. Code erstellen
3a. play.php

Die play.php Datei beinhaltet den Code, um das Quiz spielen zu können.

Hierfür wird anhand der an die play.php Datei übergebene quiz_id die Spalte „quiz_json“ aus der Datenbank ausgelesen, in der alle Daten über das Quiz als String stehen, z.B. der Quizname, Quiz-Ersteller, der Fragetype, also ob es sich um eine Freitextfrage, eine DropDown Frage oder um eine Multiple-Choice Frage handelt. Ebenso ist die Quizfrage und die richtige Antwort enthalten, sowie die Gesamtpunktzahl des Quizzes. Der Inhalt des Strings „quiz_json“ wird in der play.php Datei in das JSON-Format gebracht, um mit der PHP-Objektnotation auf die einzelnen Inhalte wie z.B. den Quiznamen zugreifen zu können. Anschließend wird die Webseite dynamisch anhand der Inhalte aus dem JSON-Objekt aufgebaut. Zuerst wird der Quizname ausgegeben und anschließend werden die Antwortmöglichkeiten, anhand des Quiz-Typen ausgegeben.

Wenn alle Frage beantwortet sind kann man auf den Button „Jetzt Antwort prüfen“ klicken und man gelangt auf die Auswertungsseite (check.php). Die ausgewählten Antworten werden im POST-Array an das check.php gesendet und dort ausgewertet.

Code:

```
<html>
<head>
   <!-- Bottstrap einbinden -->
   <meta charset="utf-8"/>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">
   <link href ="../Design/header.css" rel="stylesheet">
   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>
</head>
<body>

<?php
//Session, um auf User-ID zugreifen zu können (Wer ist angemeldet?)
session_start();
$user_id = $_SESSION['user_id'];

//Einbinden der Konfig-Datei, die die einzelnen Klassen der Fragen enthält
require_once '../config/config.php';

//Code für Kopfzeile
```

```
Helper::printHeader();

?>

<div class="container">
   <div class="row align-items-center">
      <div class="col-9">
         <br>
         <h2 class="display-2">Fragen</h2>
         <br>
         <?php

         //Quiz_ID aus GET_Request holen, die beim Klick auf "Quiz-Spielen" die Quiz_ID übermittelt
         $quiz_id = $_GET['quiz_id'];

         //DB-Abfrage mit obiger Quiz_ID, um die Daten im JSON-String-Format zu holen:
         $mysqli = new mysqli("localhost", "root", "", "baq");

         // DB-Verbindung prüfen
         if ($mysqli === false)
         {
            //Falls Verbindung fehlschlägt -> Abbrechen
            die("ERROR: Could not connect. " . $mysqli->connect_error);
         }
         else
         {
            //sonst -> hole die Daten aus der DB
            $select = "select quiz_json from quiz where quiz_id=$quiz_id";
            $result = $mysqli->query($select);
            $result = $result->fetch_assoc();
            $daten = $result['quiz_json']; //in $daten liegt jetzt der String aus der DB
         }

         // DB-Verbindung schließen
         $mysqli->close();

         //String in JSON-Fromat umwandeln
         $json_daten = json_decode($daten);

         //zu Testzwecken die Daten ausgeben
         // print_r($json_daten);

         //Fragen mit Inhalt aus DB($json_daten) und den vorgefertigten Frage-KLassen zusammenbauen:
         ?>
         //Formular für Eingabedaten
         <form action="check.php" method="post">

            <?php
            //$fragen_counter wird bei jeder erstellen Frage um 1 erhöht
            $fragen_counter = 0;

            //Iteriert über alle Fragen, die erstellt wurden:
            foreach($json_daten as $key => $value) //Liefert die json-Objekte = Anzahl erstellter Fragen
            {
               $fragen_counter++;
               echo "<h3>Frage: ".$fragen_counter."</h3>";

               //Fall 1: Wenn Frage-Typ "dropdown" -> erzeugte Dropdown-Frage
               if($json_daten[$key]->type == "dropdown")
               {
                  $dropDownQuestion = new DropDown();
                  $dropDownQuestion->buildQuestion($json_daten[$key]->question,$json_daten[$key]->answers);
                  echo "<br>";
               }
               //Fall 2: Wenn Frage-Typ "multiplechoice" (Einfachauswahl) -> erzeuge MultipleChoice-Frage
               else if($json_daten[$key]->type == "multiplechoice")
               {
                  $multipleChoiceQuestion = new MultipleChoice();
                  $multipleChoiceQuestion->buildQuestion($json_daten[$key]->question,$json_daten[$key]->answers);
                  echo "<br>";
               }
               //Fall 3: Wenn Frage-Typ "multiplechoicema" (Mehrfachauswahl) -> erzeugte MultipleChoiceMA-Frage (=Checkbox)
               else if($json_daten[$key]->type == "multiplechoicema")
               {
                  $multipleChoicemaQuestion = new MultipleChoiceMA();
```

```
                $multipleChoicemaQuestion->buildQuestion($json_daten[$key]->question,$json_daten[$key]->answers);
                echo "<br>";
            }
            //Fall 4: Wenn Frage "freetext -> erzeuge Freitext-Frage
            else if($json_daten[$key]->type == "freetext")
            {
                $freeTextQuestion = new FreeText();
                $freeTextQuestion->buildQuestion($json_daten[$key]->question,$json_daten[$key]->solution);
                echo "<br>";
            }
        }
        ?>

        <div class='SubmitButtonBox, text-center'>
            //Button, um Eingabewerte an Server zu senden
            <button type="submit" class="btn btn-info btn-lg">Jetzt Antworten prüfen</button>
        </div>

        <br>

        <!--<button type="submit">Jetzt Antworten prüfen</button> -->
    </form>

    </div>
  </div>
</div>

//Code für Fußzeile
<?php Helper::printFooter(); ?>

</body>

</html>
```

3. Code erstellen
3b. check.php

In der check.php Datei findet die Auswertung des Quizzes statt, sowie die optische Anzeige auf der Webseite, die dem User ein Feedback über seine gewählten Antworten gibt, sowie seine erreichte Punktzahl.

Zuerst wird wieder eine Datenbankabfrage gemacht und die Daten aus der Datenbank geholt. Anschließend werden sie wieder ins JSON-Format gebracht. Anschließend werden die Antworten aus der play.php aus dem POST-Array ausgelesen und mit den jeweiligen Antworten aus dem JSON-Objekt verglichen. Wenn die Antwort richtig ist, wird eine Nachricht auf der Webseite

angezeigt, dass die Antwort richtig ist und auf den Zähler, der die Gesamtpunktzahl aufsummiert, wird die Punktzahl der Frage aufaddiert.

Am Ende wird die erreichte Punktzahl, sowie die maximal zu erreichende Gesamtpunktzahl ausgegeben. Anschließend wird die erreichte Punktzahl in die Datenbank eingetragen und der Spielzähler wird um eins erhöht.

Anschließend kommt man mit einem Klick auf den Button „Zur Rangliste“ weiter zur Rangliste, wo man sich mit anderen Spielern des Quizzes anhand der Punktzahl vergleichen kann.

Code:

```
<!doctype HTML>
<html>
<head>
   <meta charset="utf-8"/>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">
   <link href ="../Design/header.css" rel="stylesheet">
   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

   <!-- CSS-Angaben für optische Darstellung auf Webseite -->
   <style>
      .right{ border: green 2px solid;
         padding: 5px;
         margin: 2px;

         background-color: #F0FFF0;
      }
      .false{ border: red 2px solid;
         padding: 5px;
         margin: 2px;
         background-color: #FFA07A;
      }
      .result{
         border: black 2px solid;
         background-color: #D3d3d3;
         padding: 5px;
         margin: 2px;
      }
   </style
</head>

<?php
//Session, um auf User-ID zugreifen zu können (Wer ist angemeldet?)
session_start();
$user_id = $_SESSION['user_id'];

//Einbinden der Konfig-Datei, die die einzelnen Klassen der Fragen enthält
require_once '../config/config.php';

//Code für Kopfzeile
Helper::printHeader();
?>

<body>
<div class="container">
   <div class="row align-items-center">
      <div class="col-9">

         <?php

         //Quiz_ID aus GET_Request holen, die beim Klick auf "Quiz-Spielen" die Quiz_ID übermittelt
         $quiz_id = $_GET['quiz_id'];
```

```php
//DB-Abfrage mit obiger Quiz_ID, um die Daten im JSON-String-Format zu holen:
$mysqli = new mysqli("localhost", "root", "", "baq");

// DB-Verbindung prüfen
if ($mysqli === false)
{
    //Falls Verbindung fehlschlägt -> Abbrechen
    die("ERROR: Could not connect. " . $mysqli->connect_error);
}
else
{
    //sonst -> hole die Daten aus der DB
    $select = "select quiz_json from quiz where quiz_id=$quiz_id";
    $result = $mysqli->query($select);
    $result = $result->fetch_assoc();
    $daten = $result['quiz_json']; //in $daten liegt jetzt der String aus der DB
}

// DB-Verbindung schließen
$mysqli->close();

//String in JSON-Fromat umwandeln
$json_daten = json_decode($daten);

//Lösungen aus DB-Json holen:
$counter_dropdown = 1;
$counter_freetext = 1;
$counter_radio = 1;
$counter_frage_mc = 0;
$counter_fragen_mcma = 1;
$counter_gesamtpunktzahl = 0;

//Alle Fragen durchgehen und Eingabewerte des Users mit Lösung aus DB Vergleichen:
$punkte = 0;
$fragen_counter = 0;

foreach($json_daten as $key => $value)
{
    $fragen_counter++;

    //Lösung für die Frage in $loesung speichern
    $loesung = $json_daten[$key]->solution;

    //Wenn Frage vom Type "dropdown"
    if($json_daten[$key]->type == "dropdown")
    {
        if(isset($_POST['DropDown'.$counter_dropdown]))
        {
            //Falls DroptDown-Frage richtig ist:
            if($loesung == $_POST['DropDown'.$counter_dropdown])
            {
                echo "<div class='right'>";
                echo "Frage: ".$fragen_counter."; ".$json_daten[$key]->question."<br>";
                echo "Richtig! <br>";
                //"Score" hochzählen
                $punkte+=($json_daten[$key]->points);
                echo "</div>";
            }
            else
            {
                //Falls DroptDown-Frage falsch ist:
                echo "<div class='false'>";
                echo "Frage: ".$fragen_counter."; ".$json_daten[$key]->question."<br>";
                echo "Falsch! <br><br>";
                $solution = $json_daten[$key]->solution;
                echo "Die Richtige Antwort wäre gewesen: \"" .$json_daten[$key]->answers[$solution]."\"";
                echo "</div>";
            }

            $counter_dropdown++;

            //Gesamtpunktzahl hochzählen
            $counter_gesamtpunktzahl += ($json_daten[$key]->points);
        }
```

```
        }
        //Wenn Frage vom Type "freetext"
        else if($json_daten[$key]->type == "freetext")
        {
            if($loesung == $_POST['Auswahl_Freetext'.$counter_freetext])
            {
                //Falls Freitext-Frage richtig ist:
                echo "<div class='right'>";
                echo "Frage: ".$fragen_counter.": ".$json_daten[$key]->question."<br>";
                echo "Richtig! <br>";
                //"Score" hochzählen
                $punkte+=($json_daten[$key]->points);
                echo "</div>";
            }
            else
            {
                //Falls Freitext-Frage falsch ist:
                echo "<div class='false'>";
                echo "Frage: ".$fragen_counter.": ".$json_daten[$key]->question."<br>";
                echo "Falsch! <br><br>";
                $solution = $json_daten[$key]->solution;
                echo "Die Richtige Antwort wäre gewesen: \"" .$solution."\"";
                echo "</div>";
            }

            $counter_freetext++;
            //Gesamtpunktzahl hochzählen
            $counter_gesamtpunktzahl += ($json_daten[$key]->points);
        }
        else if($json_daten[$key]->type == "multiplechoice")
        {
            if($loesung == $_POST['radio'.$counter_radio])
            {
                //Falls MultipleChoice-Frage richtig ist:
                echo "<div class='right'>";
                echo "Frage: ".$fragen_counter.": ".$json_daten[$key]->question."<br>";
                echo "Richtig! <br>";
                //"Score" hochzählen
                $punkte+=($json_daten[$key]->points);
                echo "</div>";
            }
            else
            {
                //Falls MultipleChoice-Frage falsch ist:
                echo "<div class='false'>";
                echo "Frage: ".$fragen_counter.": ".$json_daten[$key]->question."<br>";
                echo "Falsch! <br><br>";
                $solution = $json_daten[$key]->solution;
                echo "Die Richtige Antwort wäre gewesen: \"" .$json_daten[$key]->answers[$solution]."\"";
                echo "</div>";
            }

            $counter_radio++;
            //Gesamptpunktzahl hochzählen
            $counter_gesamtpunktzahl += ($json_daten[$key]->points);
        }

        else if($json_daten[$key]->type == "multiplechoicema")
        {
            //MultipleChoice-Frage (Checkbox)
            $antworten = $json_daten[$key]->answers;
            $length = count($antworten);
            //Ermittlung Anzahl der Lösungen
            $loesungen_length = count($json_daten[$key]->solution);
            $count_richtige_antworten = 0;
            $uebermittelte_antworten = 0;

            for($i = 0; $i<$length; $i++)
            {
                if(isset($_POST['checkbox'.$counter_fragen_mcma.$i]))
                {
                    $uebermittelte_antworten++;

                    for($j = 0; $j < $loesungen_length; $j++)
                    {
```

```
                    if($loesung[$j] == $_POST['checkbox'.$counter_fragen_mcma.$i])
                    {
                        //Wenn Lösung richtig ist, erhöhe $count_richtige_antworten
                        $count_richtige_antworten++;
                        break;
                    }
                    else
                    {
                        continue;
                    }
                }
            }
            else
            {
                continue;
            }
        }

        // Wenn die Anzahl an richtigen Antworten mit der Anzahl an richtigen Lösungen identisch ist
        // UND wenn die Anzahl der übermittelten Antworten ebenfalls der Anzahl an richtigen Lösungen
      entspricht
        // D.h. es müssen also 1. Alle Antworten richtig sein UND 2. es dürfen nicht zu viele/ zu wenig Antworten ausgewählt
werden
        if ($count_richtige_antworten == $loesungen_length && $uebermittelte_antworten == $loesungen_length)
        {
            //Falls MultipleChoice(Checkbox)-Frage richtig ist:
            echo "<div class='right'>";
            echo "Frage: ".$fragen_counter.": ".$json_daten[$key]->question."<br>";
            echo "Richtig! <br>";
            //"Score" hochzählen
            $punkte+=($json_daten[$key]->points);
            echo "</div>";
        }
        else
        {
            //Falls MultipleChoice(Checkbox)-Frage falsch ist:
            echo "<div class='false'>";
            echo "Frage: ".$fragen_counter.": ".$json_daten[$key]->question."<br>";
            echo "Falsch! <br><br>";
            $solution = $json_daten[$key]->solution;
            echo "Die Richtige Antwort wäre gewesen: <br>";
            foreach($solution as $value)
            {
                echo $json_daten[$key]->answers[$value]."<br>";
            }

            echo "</div>";
        }
        $counter_fragen_mcma++;
        //Gesamptpunktzahl hochzählen
        $counter_gesamtpunktzahl += ($json_daten[$key]->points);
      }
    }
    echo "<br>";
    echo "<div class='result'>";

    //Anzeigen der erreichten Punktzahl
    echo "<h4>Ihr Ergebnis</h4>";

    if($punkte == 1)
    {
        echo "Sie haben: ".$punkte . " Punkt von insgesamt: ".$counter_gesamtpunktzahl." Punkten erreicht!";
    }
    else
    {
        echo "Sie haben: ".$punkte . " Punkte von insgesamt: ".$counter_gesamtpunktzahl." Punkten erreicht!";
    }
    echo "</div>";

    //Ergebnis in Datenbank speichern:
    $mysqli = new mysqli("localhost", "root", "", "baq");

    // DB-Verbindung prüfen
    if ($mysqli === false)
    {
```

```
            die("ERROR: Could not connect. " . $mysqli->connect_error);
        }
        else
        {
            //erreichte Punktzahl eintragen
            $insert = "update score set points = $punkte where user_id='$user_id' AND quiz_id='$quiz_id'";
            $mysqli->query($insert);

            //Counter 0 bedeutet -> User hat das Spiel noch nicht gespielt und wird deshalb noch nicht in der Rangliste       angezeigt
            //ersten wenn Counter == 1 wird User angezeigt
            $update = "update score set counter = (counter +1) where user_id='$user_id' AND quiz_id='$quiz_id'";
            $mysqli->query($update);
        }
        // DB-Verbindung schließen
        $mysqli->close();

        //Weiter zur Rangliste
        ?>
        <br>
        <br>
        <br>

        //Quiz-ID wird als "Hidden-Input-Feld" an rangliste.php weitergegeben
        <!-- Quiz_ID an rangliste.php übergeben -->
        <form action="rangliste.php" method="post">
            <?php echo "<input type='hidden' name='quiz_id' value='".$quiz_id."'>"; ?>
            <div class='SubmitButtonBox, text-center'>
                <button type="submit" class="btn btn-info btn-lg">Zur Rangliste</button>
            </div>
            <br>
        </form>
    </div>
  </div>
</div>

<?php

//Code für Fußzeile
Helper::printFooter();
?>

</body>
</html>
```

3. Code erstellen
3c. rangliste.php

Die Ranglisten-Seite wird ebenfalls dynamisch mit php erstellt. Auf dieser Seite werden für ein Quiz alle Ergebnisse der Quiz-Teilnehmer, in tabellenform, dargestellt. Der Spieler mit der höchsten Punktzahl ist in der Tabelle ganz oben. Derjenige mit der niedrigsten Anzahl ist ganz am Ende der Tabelle. User, die zu einem Quiz eingeladen wurden, dieses aber noch nicht gespielt haben (Eintrag

in Tabelle Score der DB ist gesetzt, der Eintrag in der Spalte counter ist 0) werden in dieser Rangliste noch nicht angezeigt. Sie werden erst angezeigt, wenn sie das erste mal das Quiz gespielt haben. Wenn ein User das Quiz öffter spielt, wird immer sein letzter Versuch gewertet und die DB entsprechend der erreichten Punkte geupdatet, auch wenn der User in dem neuen Versuch weniger Punkte erreicht hat, als beim letzten Versuch.

Code:

```
<!doctype HTML>
<html>
<head>

   <meta charset="utf-8"/>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
         integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">
   <link href ="../Design/header.css" rel="stylesheet">
   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>

   <style>
      <!-- CSS-Angaben für optische Darstellung auf Webseite -->
      table, th{
         border-width: 1px;
         border-style: solid;
         border-color: black;
         padding: 10px;
      }

      h3 {
         margin-top: 25px;
         margin-bottom: 10px;
         text-align: center;
      }

      td{
         text-align: center;
         font-size: 24px;
      }

      th{
         text-align: center;
         font-size: 26px;
      }
   </style>

   <?php

   //Session, um auf User-ID zugreifen zu können (Wer ist angemeldet?)
   session_start();
   $user_id = $_SESSION['user_id'];

   //Einbinden der Konfig-Datei, die die einzelnen Klassen der Fragen enthält
   require_once '../config/config.php';

   //Code für Kopfzeile
   Helper::printHeader();

   //DB-Abfrage mit quiz-id, um Ergebnisse vom Quiz zu holen
   //Quiz-ID wird über $_POST (hidden-input Feld) übergeben
   $quiz_id = $_POST['quiz_id'];

   //DB-Abfragen, um Daten vom Quiz zu holen
   $mysqli = new mysqli("localhost", "root", "", "baq");
```

```
    // DB-Verbindung prüfen
    if ($mysqli === false)
    {
        die("ERROR: Could not connect. " . $mysqli->connect_error);
    }
    else
    {
        //Quiz-Name aus DB holen
        $quiz_name = "select quiz_name from quiz where quiz_id=$quiz_id";
        $quiz_name = $mysqli->query($quiz_name);
        $res = $quiz_name->fetch_assoc();

        echo "
                </head>
                <body>

                <div class='container'>
                <div class='row align-items-center'>
                <div class='col-9'>

                <h3>Rangliste: \"".$res['quiz_name']."\" (Quiz-ID: ".$quiz_id.")</h3>
                <table class='table'>
                <th>Profilbild</th> <th>Username</th> <th>Punkte</th>
                ";

        //Alle User aus DB holen, die das Quiz gespielt haben
        $select = "select u.username, u.user_id, r.points, r.counter from users u inner join score r on u.user_id = r.user_id order by r.points desc";
        $result = $mysqli->query($select);

        //Alle Einträge der Spalte counter holen -> wird zur Prüfung benötigt, ob User in Rangliste angezeigt wird, wenn counter >0 ist
        $versuche = "select counter from score where quiz_id=$quiz_id";
        $versuche = $mysqli->query($versuche);


        while ($row = $result->fetch_assoc())
        {
            //Anzeige in Rangliste nur, wenn das Quiz mind. 1x gespielt wurde
            if($row['counter'] > 0 && $row['counter'] != "")
            {
                echo "<tr>";

                //Profilbild einbinden
                if(file_exists('C:/xampp/htdocs/BAQ/PHP-Code/upload/'.$row["user_id"].'.jpg'))
                {
                    echo "<td><img height='50px' src='/BAQ/PHP-Code/upload/".$row['user_id'].".jpg'></td>";
                }
                else if(file_exists('C:/xampp/htdocs/BAQ/PHP-Code/upload/'.$row["user_id"].'.png'))
                {
                    echo "<td><img height='50px' src='C:/xampp/htdocs/BAQ/PHP-Code/upload/".$row['user_id'].".png'></td>";
                }
                else
                {
                    echo "<td><img height='50px' src='/BAQ/PHP-Code/upload/BAQ.PNG'></td>";
                }
                //echo "<td><img height='50px' src='/BAQ/PHP-Code/upload/".$row['user_id'].".jpg'</td>";
                echo "<td>".$row['username']."</td>";
                echo "<td>".$row['points']."</td>";
                echo "</tr>";
            }
        }
    }
    // DB-Verbindung schließen
    $mysqli->close();
    ?>
    </table>
    </div>
    </div>
    </div>

    <!-- Code für Fußzeile -->
    <?php Helper::printFooter(); ?>
    </body>
</html>
```