Stefan Waidele
Ensisheimer Straße 2
79395 Neuenburg am Rhein
Stefan.Waidele@AKAD.de

AKAD Hochschule Stuttgart
Immatrikulationsnummer: 102 81 71

Modul DBA02 — Praktisches Arbeiten mit Datenbanken
Assignment

Datenbankgestützte PHP–Anwendung für eine Umfrage-Website

Betreuer: Prof. Dr. Franz–Karl Schmatzer

13. Oktober 2013

AKAD Hochschule Stuttgart

Inhaltsverzeichnis

Abbildungsverzeichnis **ii**

1 Einleitung **1**
1.1 Aufgabenstellung 1
1.2 Gemeinschaftsarbeit 2
1.3 Aufbau der Arbeit 2
1.4 Abgrenzung 3

2 Grundlagen **4**
2.1 Das Entity–Relationship Modell 4
2.2 Entwurfsmuster: Fassade 4
2.3 Entwurfsmuster: Singleton 4
2.4 PHP–Schnittstelle — Session–Cookies 5
2.5 PHP–Schnittstelle — PDO 6
2.6 HTML–Designframework — Bootstrap 6

3 Datenbank–Schema **6**
3.1 Tabelle: user 6
3.2 Tabelle: frage 7
3.3 Tabelle: antwort 7
3.4 Tabelle: geantwortet 8
3.5 Query: Frage – Text 8
3.6 Query: Antwortmöglichkeiten 8
3.7 Query: Anzahl der gegebenen Antworten 8
3.8 Query: Gegebene Antworten und Häufigkeit 9

4 Hauptteil **10**
4.1 Tabelle 10
4.2 Bilder 10
4.3 Syntax Highlighting 10

5 Bewertung **12**
5.1 Zusammenfassung 12
5.2 kritische Würdigung 12
5.3 Ausblick 12
5.4 Erfolgsfaktoren 12

Literatur **iii**

Abbildungsverzeichnis

1 SQL: CREATE TABLE user 7
2 SQL: CREATE TABLE frage 7
3 SQL: CREATE TABLE antwort 8
4 SQL: CREATE TABLE geantwortet 8
5 SQL: Text von Frage <n> 8
6 SQL: Mögliche Antworten für Frage <n> 9
7 SQL: Antwortzahl 100% von Frage <n> 9
8 SQL: Gegebene Antworten mit Häufigkeit für Frage <n> 9
9 Akad 10
10 Quellcode: Aufruf von header.php (PHP) 11

1 Einleitung

1.1 Aufgabenstellung

Im Rahmen des Moduls DBA02 war eine datenbankgestützte PHP–Anwendung für eine Website zu erstellen, welche die folgenden Kriterien erfüllt:

- **Frage stellen**: Besuchern der Website soll eine Frage gestellt werden, auf die sie mit einer oder mehreren vorgegebenen Möglichkeiten antworten können.
- **Auswertung**: Nach der Beantwortung der Frage soll dem Besucher eine Auswertung der bisher gegebenen Antworten (Angaben in Prozent) gezeigt werden.
- **Benutzerverwaltung**: Ein Administrator soll sich bei der Anwendung anmelden können. Hierzu soll ein Benutzername und Passwort abgefragt und geprüft werden.
- **Neue Fragen eingeben**: Dem Seitenadministrator soll es über ein Formular möglich sein, neue Fragen mit den zugehörigen Antwortmöglichkeiten einzugeben. Normalen Besucher der Website ist diese Möglichkeit zu verwehren.
- **Datenbank**: Alle benötigten Daten werden in einer MySQL–Datenbank gespeichert.
- **Echtzeitstatistiken**: Die Auswertung der gegebenen Antworten soll unmittelbar vor der Anzeige berechnet werden.
- **XAMP**: Die Anwendung soll mit der Kombination von Apache–Webserver, MySQL–Datenbank und PHP als Programiersprache lauffähig sein. Das Betriebssystem kann frei gewählt werden.

Desweiteren sollte die Anwendung objektorientiert Programmiert werden, Enturfsmuster verwenden und die HTML–Ausgabe per CSS formtiert werden.

1.2 Gemeinschaftsarbeit

Die Aufgabe war arbeitsteilig in Teamarbeit zu lösen. Das der Anwendung zu Grunde liegende Datenmodell wurde gemeinsam in einer Teambesprechung erarbeitet und festgelegt. Anschließend wurde ein Mockup[1] der HTML–Seiten und die benötigten SQL–Abfragen, gefolgt von einem prozedural programmierten Prototypen erstellt. Hierbei erstellte der Autor die für die Benutzerverwaltung und Frageneingabe notwendige Programmteile. Die Abfrage– und Auswertungsseiten wurden von Yvonne Frezel gefertigt.

Da die im Seminar DBA02 begonnene Umsetzung des Programmcodes in Klassen unterschiedliche Richtungen verfolgte, wurde die enge Teamarbeit anschließend nicht mehr weitergeführt. Aufgrund der Gemeinsamen Datenbasis sind die vom Author und von Frenzel erstellten PHP–Dateien miteinander kombinierbar, auch wenn sie intern andere Klassen und Zugriffsmethoden nutzen.

Dieses Assignment geht hauptsächlich auf die vom Autor konzipierten Programmteile „Benutzerverwaltung“ und „Neue Fragen hinzufügen“ ein.

1.3 Aufbau der Arbeit

Zunächst wurden die für die Anwendungen relevanten Konzepte, Techniken, und Frameworks beschrieben. Anschließend erfolgte die Beschreibung der Implementierungsdetails und der vom Autor gewählten Lösungsmöglichkeiten

[1] engl. für Attrappe. (to mock: nachahmen)

1.4 Abgrenzung

Da eine Datenbankgestützte Web–Anwendung in der Regel einem großen Personenkreis[2] zur Verfügung steht sind hier unbedingt Sicherheitsaspekte zu beachten. Da eine ausführliche Betrachtung dieser Maßnahmen den Rahmen dieses Dokuments sprechengen würde, werden nur entsprechende Hinweise auf weiterführende Informationen gegeben. Auch Performance–Überlegungen gehen nur in sehr beschränktem Maß in die Implementation ein.

[2]allen Internet– oder zumindest Intranetnutzern

2 Grundlagen

2.1 Das Entity–Relationship Modell

2.2 Entwurfsmuster: Fassade

Beim Fassaden–Entwurfsmuster gewährt eine Klasse einen einfachen Zugriff auf ein beliebig komplextes System weiterer Klassen. Den Nutzer der Fassadenklasse benötigt kein Wissen über die Funktionsweise der Klassenhierarchie hinter der Fassade, kann jedoch auf diese zugreifen, falls die bereitgestellte Funktionalität nicht ausreicht[3].

In der hier erstellten Anwendung wird der Zugriff auf die Datenbank über die Fassaden–Klasse `SQL` realisiert. Diese erstellt das Low–Level Obkekt der Klasse `Datenbank`, bereitet die notwendigen SQL–Abfragen vor und gibt die Resultate dann als String oder als Array von Strings zurück. Die Aufrufenden Routinen benötigen kein Wissen über die verwendete Datenbankschnittstelle oder über die Details der Abfragen. Sollten die in der Klassendefinition vorgesehenen Abfragen allerdings nicht ausreichen, kann auch direkt auf die Klasse Datenbank zugegriffen werden.

2.3 Entwurfsmuster: Singleton

Das Klasse nach dem Singleton–Entwurfsmuster stellt sicher, dass es in einem Programm von einer Klasse nur ein einziges Mal instanziiert wird. Allen Nutzern der Klasse wird dann eine Referenz auf ebendiese Instanz übergeben, der Zugriff erfolgt jeweils auf die gleichen Daten, die somit global zur Verfügung gestellt werden[4].

[3] vgl. [Balzert, 2005], Seite 367ff
[4] vgl. [Balzert, 2005], Seite 361ff

Somit bildet das Singleton–Designpattern eine passende Grundlage für die Nutzerverwaltung, da immer nur ein Benutzer angemeldet sein kann[5].

Das Singleton-Entwurfsmuster kann aufgrund seiner Eigenschaften als objektorientierte Umsetzung von globalen Variablen mit all deren Vor– und Nachteilen gesehen werden und wird daher auch als „Anti–Pattern“ kritisiert[6]

2.4 PHP–Schnittstelle — Session–Cookies

Da HTTP ein zustandsloses Protokoll ist, wird ein Mechanismus benötigt, mit dem gespeichert werden kann, ob es sich beim Besucher der Website um einen angemeldeten Benutzer handelt oder nicht. Die von PHP bereitgestellten Session–Cookies können eine begrenzte Menge Daten (ca. 4kB), die im Browser gespeichert wird, von Seitenaufruf zu Seitenaufruf weitergeben[7].

Die hier besprochene Anwendung verwedet diese Möglichkeit um den Anmeldestatus (`angemeldet==TRUE` bzw. `angemeldet==FALSE`) und den Benutzernamen zu speichern.

[5] Dies gilt jeweils pro Browser–Instanz. In einem weiteren Browserfenster mit eigenen Cookies kann sich ein weiterer Nutzer anmelden, jedoch auch wieder nur einer
[6] vgl. [Hauer, 2010]
[7] Vgl. [Theis, 2013], S. 417ff

2.5 PHP–Schnittstelle — PDO

2.6 HTML–Designframework — Bootstrap

3 Datenbank–Schema

3.1 Tabelle: user

user		
email	varchar(255)	
pw	char(32)	

user	

Tabelle 1: Beschreibung. Quelle: Berger, Vorlesung, 2012, München

```
CREATE TABLE user (
  email VARCHAR(255) NOT NULL,
  pw CHAR(32) NOT NULL,
  create_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (‘email‘));
```

Abbildung 1: SQL: CREATE TABLE user

Als Benutzername wird hierbei die E–Mail Adresse des Administrators genutzt.

Das Passwort sollte nicht im Klartext in der Datenbank gespeichert werden. Ein `salted hash`[8] schützt hier das Passwort vor dem Ausspähen durch den Administrator selbst[9] oder durch Angreifer.
Die hier reservierten 32 Byte sind für den in der MySQL–Dokumentation[10] emphohlenen MD5-Hash ausreichend. Da die entsprechende PHP–Dokumentation[11] hier allerdings eine genau entgegengesetzte Empfehlung gibt, ist dieser Sicherheitsaspekt für ein Produktivsystem nochmals genauer zu prüfen.

3.2 Tabelle: frage

```
CREATE TABLE frage (
  fid INT NOT NULL AUTO_INCREMENT,
  txt VARCHAR(1024) NOT NULL,
  PRIMARY KEY (‘fid‘));
```

Abbildung 2: SQL: CREATE TABLE frage

3.3 Tabelle: antwort

[8] Also der Hashwert des Passworts, welches zuvor mit Applikationsspezifischen Zusatzdaten ergänzt wurde
[9] Seit Vodafone 2013 eine dokumentierte Gefahr
[10] [**?**]
[11] [**?**]

```
CREATE TABLE antwort (
  aid INT NOT NULL AUTO_INCREMENT,
  nr  INT NULL,
  txt VARCHAR(1024) NOT NULL,
  fid INT NOT NULL,
  PRIMARY KEY (`aid`),
  FOREIGN KEY (`fid`) REFERENCES frage(`fid`));
```

Abbildung 3: SQL: CREATE TABLE antwort

3.4 Tabelle: geantwortet

```
CREATE TABLE geantwortet (
  gid INT NOT NULL AUTO_INCREMENT,
  aid INT NOT NULL,
  zs TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`gid`),
  FOREIGN KEY (`aid`) REFERENCES antwort(`aid`));
```

Abbildung 4: SQL: CREATE TABLE geantwortet

3.5 Query: Frage – Text

Wird für die Abfrage und Auswertung benötigt.

```
select txt from frage where fid = <n>;
```

Abbildung 5: SQL: Text von Frage <n>

3.6 Query: Antwortmöglichkeiten

Wird für die Abfrage benötigt.

3.7 Query: Anzahl der gegebenen Antworten

Wird für die Auswertung benötigt: 100%

```
select antwort.txt
        from antwort
        where fid = <n>
```

Abbildung 6: SQL: Mögliche Antworten für Frage <n>

```
select count(geantwortet.aid)
        from antwort, geantwortet
        where geantwortet.aid = antwort.aid and antwort.fid = <n>;
```

Abbildung 7: SQL: Antwortzahl 100% von Frage <n>

3.8 Query: Gegebene Antworten und Häufigkeit

Wird für die Auswertung benötigt.

```
select antwort.txt, count(geantwortet.aid)
        from antwort, geantwortet
        where geantwortet.aid = antwort.aid and antwort.fid = <n>
        group by geantwortet.aid;
```

Abbildung 8: SQL: Gegebene Antworten mit Häufigkeit für Frage <n>

4 Hauptteil

4.1 Tabelle

Head1	Head2	Head3
1	2	3
4	5	6
7	8	9
1	2	3
4	5	6
7	8	9

Tabelle 2: Beschreibung. Quelle: Berger, Vorlesung, 2012, München

4.2 Bilder

Abbildung 9: Akad. Quelle: www.akad.de

4.3 Syntax Highlighting

```php
<?php
$title="Lorem";
$desc = "Lorem Ipsum";
include($_SERVER['DOCUMENT_ROOT'].'/header.php');
?>
```

Abbildung 10: Quellcode: Aufruf von header.php (PHP)

5 Bewertung

5.1 Zusammenfassung

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

5.2 kritische Würdigung

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

5.3 Ausblick

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

5.4 Erfolgsfaktoren

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Literatur

[Balzert, 2005] Balzert (2005). *Lehrbuch der Objektmodellierung*. Spektrum Akademischer Verlag, München.

[Hauer, 2010] Hauer (2010). Das Singleton Design Pattern, Abruf am 11.10.2013. `http://www.philipphauer.de/study/se/design-pattern/singleton.php#nachteile`.

[Strickel, 1991] Strickel (1991). *Datenbankdesign: Methoden und Übungen*. Gabler, Wiesbaden.

[Theis, 2013] Theis (2013). *Einstieg in PHP 5.5 und MySQL 5.6*. Gabler, Wiesbaden.

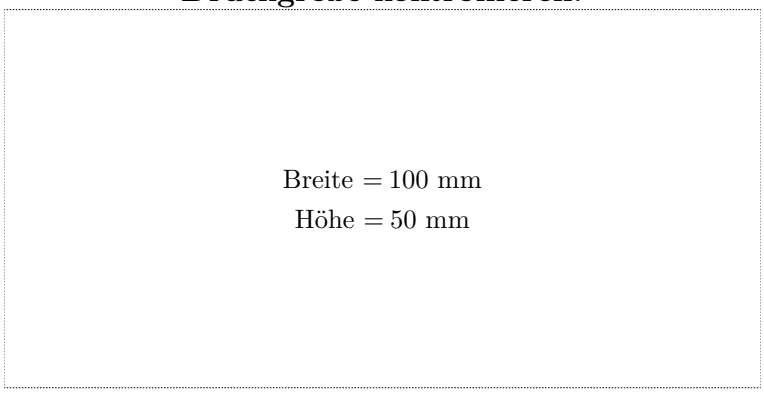
Eidesstattliche Erklärung

Ich versichere, dass ich das beiliegende Assignment selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie alle wörtlich oder sinngemäß übernommenen Stellen in der Arbeit gekennzeichnet habe.

_____ _____

(Datum, Ort) (Unterschrift)

— Druckgröße kontrollieren! —



— Diese Seite nach dem Druck entfernen! —