

# Klausur „Entwicklung webbasierter Anwendungen, Wintersemester 2022/23

Thomas Hofmann

## Hinweise:

- Bearbeitungszeit: 90 Minuten + 10 Minuten für Down- und Upload. Insgesamt gibt es 100 Punkte in 5 Aufgaben.
- **Abgabe bis 10.02.2023, 11:55 über Git in Ihrem persönlichen Repository für die Prüfung**
- Erlaubte Hilfsmittel: Ein Notebook pro Person, das Internet, sowie alle auf dem Notebook mitgebrachten Informationen und jede auf Papier gedruckte Quelle.
- **Kommunikation mit anderen Menschen, außer der Prüfungsaufsicht, ist nicht erlaubt! Die Verwendung von Chats o.ä. wird als Täuschungsversuch gewertet.**
- Lesen Sie zuerst die gesamte Aufgabenstellung bevor Sie mit der Lösung der Aufgaben beginnen.
- Bearbeiten Sie die Aufgabe im Ordner src/Exam des Repositories, das ihnen zugewiesen wurde. Legen Sie keine neuen Dateien an und benennen Sie die Dateien nicht um. Commiten bzw. pushen Sie ihren Code via Git regelmäßig (ca. alle 10 Minuten) in Ihr zugeteiltes Repository. Die Abgabe erfolgt ausschließlich über Git. Commits mit einem Zeitstempel nach Ablauf der Bearbeitungszeit können nicht berücksichtigt werden.
- **Es wird nur bewertet, was sich in dem für Sie angelegten Gitlab-Repository in den vorbereiteten Dateien bis zum Abgabetermin unter src/Exam befindet.**
- Schreiben Sie standardkonformen Code, der die Regeln der Veranstaltung für ordentlichen Code und die Vorgaben durch die Seitenklassen berücksichtigt.
- Diese Aufgabenblätter werden nicht abgegeben. Schreiben Sie also keine Lösungsteile darauf. Sie dürfen die Seiten natürlich auch trennen.

## Zulieferung

In Ihrem GitLab-Projekt sind die Dateien, die Sie bearbeiten sollen, schon angelegt:

- Im Moodlekurs der Veranstaltung finden Sie eine ZIP-Datei mit vorbereiteten Inhalten für diese Dateien sowie ein SQL-Skript zum Erzeugen und Füllen der Datenbank. Kopieren Sie zunächst diese Dateien in das Verzeichnis src/Exam und überschreiben Sie die vorhandenen Dateien.
- Wenn Sie die Datenbank mit dem SQL-Skript importieren wollen melden Sie sich im phpmyadmin mit dem **Benutzer: root** und **Passwort: ganzGeheim** an. Anschließend nutzen Sie die „Importieren“ Funktionalität, wie wir es kennengelernt haben.
- Page.php: Die bekannte Basisklasse der Seitenklassen (Sie müssen diese Datei nicht mehr bearbeiten!)
- Exam.php – siehe Aufgabe 1
- ExamAPI.php – siehe Aufgabe 2
- Exam.js – Siehe Aufgabe 3
- Exam.css – siehe Aufgabe 4

## Die Aufgabe allgemein: „HDA\_Chatbot“

Da dieses Jahr Chatbots in aller Munde sind, sollen Sie einen eigenen Chatbot programmieren. Dieser soll sehr simpel sein und auf eine kleine Anzahl an Fragen antworten können.

Das Interface vom Chatbot beinhaltet neben einer Navigation, eine Liste mit allen Fragen welche man dem Chatbot stellen kann. Außerdem gibt es ein Eingabefeld und einen Knopf zum Fragen stellen.

Wurde eine Frage gestellt, antwortet der Chatbot in der Chatausgabe. Die vordefinierten Fragen und Antworten werden aus der Datenbank entnommen.

## Visuelle Darstellung

**HDA\_Chatbot**

[Home](#) [Impressum](#) [Datenschutz](#)

**Folgendes kann man mich fragen**

1. Wer bist du?
2. Wie alt bist du?
3. Was ist dein Lieblingessen?
4. Was ist dein Lieblingsfach?
5. Was ist deine Meinung zu Informatik?

**Chatausgabe**

17:37:27: Ich bin HDA\_Chatbot!

17:37:30: Tut mir leid, auf diese Frage habe ich keine Antwort.

Meine Frage ...

Copyright © Thomas Hofmann

Abbildung 1: Chatbot Interface mit beispielhafter Chatausgabe

## Datenbankstruktur

Die Datenbank besteht aus einer Tabelle namens interaction. Diese ist wie folgt aufgebaut:

id	question	answer
1	Wer bist du?	Ich bin HDA_Chatbot!
2	Wie alt bist du?	Ich bin noch gar nicht geboren!
3	Was ist dein Lieblingessen?	Alles was der Infotreff anzubieten hat!
4	Was ist dein Lieblingsfach?	EWA!
5	Was ist deine Meinung zu Informatik?	Informatik ist super!

id	question	answer
1	Wer bist du?	Ich bin HDA_Chatbot!
2	Wie alt bist du?	Ich bin noch gar nicht geboren!
3	Was ist dein Lieblingessen?	Alles was der Infotreff anzubieten hat!
4	Was ist dein Lieblingsfach?	EWA!
5	Was ist deine Meinung zu Informatik?	Informatik ist super!

## Aufgabe 1: HTML/PHP (25 Punkte)

Implementieren Sie in der Datei **Exam.php** unter Verwendungen der **Seitenklassenstruktur**, sowie **PHP** und **HTML**, die in der Abbildung 1 dargestellte Webseite. Die Seite hat folgende Struktur:

- Einen Kopfteil mit einer Hauptüberschrift und einer Navigation samt Links.
- Einen Hauptteil mit folgenden Elementen:
  - Eine Liste, welche dynamisch aus den Daten der Datenbanktabelle: interaction generiert wird. Diese Liste beinhaltet alle Fragen, welche man dem Chatbot stellen kann. Die Erzeugung der Liste sollte gegen das Cross-Site-Scripting abgesichert sein.
  - Eine Chatausgabe. Dieses Element dient als Ausgabe für die spätere Interaktion mit dem Chatbot und soll mit JavaScript angesprochen werden können.
  - Ein Eingabefeld. Dieses soll ebenfalls mit JavaScript angesprochen werden können.
  - Ein Knopf welcher später die eingegebene Frage beim Klicken via JavaScript abschicken soll.
- Ein Fußteil mit dem Namen des Programmierers.

### Anmerkungen

- Es sollen so gut es geht passende HTML-Elemente verwendet werden.
- Die Links der Navigation müssen keine funktionsfähige Weiterleitung haben.
- Auch wenn das so aussieht, sollen die Daten nicht via HTML-Formular abgeschickt werden. Dies geschieht später in Aufgabe 3 via JavaScript/Ajax.
- Die Chatausgabe ist kein Input-Element, sondern ein ganz normaler Container. Das Aussehen soll später via CSS realisiert werden.

## Aufgabe 2: PHP (25 Punkte)

Implementieren Sie in der Datei **ExamAPI.php** unter Verwendung der **Seitenklassenstruktur** ein **PHP-Skript** welches als Response ein **JSON-String** zurückliefert. Folgendes ist zu beachten:

- Im späteren Verlauf (Aufgabe 3) soll via JavaScript ein Ajax-Request an dieses PHP-Skript geschickt werden. Dieser Request beinhaltet die gestellte Frage an den Chatbot in Form eines GET-Parameters. Das für diese Aufgabe zu programmierende PHP-Skript soll als Response die dazugehörige Antwort aus der Datenbanktabelle interaction als JSON-String zurückliefern.
- Beispiel: Der Request **ExamAPI.php?Question=Wer%20bist%20du?** liefert den Response „**Ich bin HDA\_Chatbot!**“ als JSON-String zurück. Siehe Inhalt der Datenbank.
- Dieses Skript sollte bestenfalls, auch bei falschen/manipulierten GET-Parametern, keine Fehler ausspucken.
- Außerdem sollte es gegen jegliche SQL-Injections abgesichert werden.

### Anmerkungen

- Denken Sie daran sich sehr akkurat an die vorgegebenen Strukturen der Seitenklassen zu halten. Jede in der Vorlesung kennengelernte Klassenmethode hat hier ihre Aufgabe.
- Die **%20** in der URL bedeutet ein kodiertes Leerzeichen.
- Sie können die Funktionalität ihrer Lösung testen indem Sie im Browser die entsprechende URL samt GET-Parameter aufrufen. Wie oben im Beispiel gezeigt. Bekommen Sie die richtige Antwort?

### Aufgabe 3: JavaScript (25 Punkte)

Implementieren Sie in der **Exam.js** Datei via **JavaScript** ein Ajax-Request, welcher an **Exam22API.php** abgeschickt wird. Fortan sollen die angefragten Daten verarbeitet und mittels **HTML** in der Chatausgabe visualisiert werden. Folgendes ist zu beachten:

- Der Request soll abgeschickt werden, wenn auf den Knopf „Frage abschicken“ geklickt wird.
- Mit dem Request zusammen soll als GET-Parameter das Value des Frage-Eingabefeldes mitgeschickt werden.
- Die mit dem Ajax-Request angefragten Daten sollen schließlich verarbeitet und via JavaScript in die Chatausgabe eingefügt werden.
- Jede Antwort sollte zusätzlich bei der Ausgabe mit der aktuellen Uhrzeit versehen sein. Siehe Abbildung 1.
- Neue Antworten werden unter die vorherigen Antworten angehängt. Siehe Abbildung 1.
- Wurde keine passende Antwort zurückgeliefert soll „Tut mir leid, auf diese Frage habe ich keine Antwort.“ in der Chatausgabe erscheinen. Um das zu realisieren dürfen Sie, falls benötigt, ihr PHP-Skript aus der Aufgabe 2 nochmal anpassen.

#### Anmerkungen

- Mit z.B. **let date = new Date();** bekommen Sie das JavaScript Date Objekt. Dieses beinhaltet Methoden um die aktuelle Uhrzeit auszugeben. Andere Vorgehensweisen sind auch in Ordnung.
- Die gestellte Frage soll nicht in der Chatausgabe erscheinen um das Design minimalistisch zu halten.

### Aufgabe 4: CSS (15 Punkte)

Implementieren Sie in der **Exam.css** Datei via **CSS** das Styling der Webseite. Die **Abbildung 1** sollte Ihnen als Vorlage dienen. Folgendes soll realisiert werden:

- Die ganze Seite hat die Schrift Verdana, sans-serif.
- Der Hintergrund vom Chatbot-Interface hat die Farbe lightgray. Außerdem besitzt er eine schwarze Umrandung und runde Ecken.
- Die Hauptüberschrift ist mittig zentriert.
- Die Links der Navigation sind mittig zentriert und nebeneinander.
  - Ab einer Bildschirmbreite kleiner 600px sind die Links untereinander.
- Die Chatausgabe hat eine Höhe von mindestens 100px, der Hintergrund ist weiß und die Schrift kursiv. Außerdem hat sie eine schwarze Umrandung und runde Ecken.
- Das Verhalten vom Eingabefeld und dem Knopf soll mit Hilfe von Flexbox realisiert werden.
  - Die Elemente sollen in einer Reihe sein.
  - Das Verhältnis von Eingabefeld zu Knopf soll 4 zu 2 sein.
  - Kann dieses Verhältnis nicht mehr erfüllt werden, sollen die Items untereinander springen.
- Der Text von der Fußzeile ist mittig zentriert.
- Versuchen Sie die Abstände ungefähr so wie in der Abbildung 1 zu realisieren.

### Aufgabe 5: Lauffähigkeit/Vollständigkeit (10 Punkte)

Gehen Sie diese Aufgabe erst an, wenn Sie die übrigen Aufgaben nach bestem Wissen gelöst haben.

Prüfen Sie die Lauffähigkeit ihrer Anwendung indem Sie das Docker-Setup starten und im localhost ihre Anwendung begutachten. Folgendes soll geprüft werden:

- Prüfen Sie ob Ihr erzeugter HTML-Code valide und barrierefrei ist.
- Sieht der Chatbot ungefähr so aus wie in der Abbildung 1?
- Vergewissern Sie sich ob die Anwendung funktionsfähig ist indem Sie dem Chatbot eine Frage stellen. Bekommen Sie die richtige Antwort?

### Abgabe

Laden Sie Ihr Endergebnis in Git hoch und prüfen Sie über GitLab (<https://code.fbi.h-da.de>), ob die Dateien auch wirklich angekommen sind.