Leistungsnachweis "Entwicklung Web-basierter Anwendungen" Wintersemester 2021/2022

Aufgabe	1	2	3	4	5	Gesamt
Punkte	24	12	16	22	6	80

Vorbereitung:

- Sie haben die Eigenständigkeitserklärung ausgefüllt, unterschrieben und in Git hochgeladen!?
- **Ohne ausgefüllte Eigenständigkeitserklärung wird Ihre Abgabe nicht gewertet und kann als Fehlversuch gewertet werden!**

Zulieferung:

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

- Page.php: Die bekannte Basisklasse der Seitenklassen mit der Einbindung von CSS und JS und mit der Datenbankanbindung. Diese Datei müssen Sie nicht bearbeiten, sondern nur einbinden.
- Exam.php – für Aufgabe 1
- 2021_WS.sql – zum Anlegen der DB (Zulieferung per Email)
- Exam.js – für Aufgabe 2 und 3
- RatingsAPI.php – Testdatei für Aufgabe 3 (Zulieferung per Email)
- Exam.txt – für Aufgabe 4
- Exam.css – für Aufgabe 5
- ExamAPI.php – wird nicht benötigt

Hinweise:

- Bearbeitungszeit: 90+15 Minuten, **Abgabe bis 21.02.2022, 17:30 Uhr über Git in Ihrem persönlichen Repository. Abgaben nach Ende der offiziellen Prüfungszeit werden mit 5 bewertet.**
- Tipp: Stellen Sie sich einen Wecker auf 5 Minuten vor der Abgabezeit!
- Erlaubte Hilfsmittel: Jede elektronische oder auf Papier gedruckte Quelle. Kommunikation (physisch oder elektronisch via Discord, Teamspeak etc) mit anderen Menschen außer der Prüfungsaufsicht ist nicht erlaubt!
- **Der Austausch von Code-Teilen führt automatisch bei allen Beteiligten zu einem Nichtbestehen und zieht die Meldung eines Täuschungsversuchs an das Prüfungsamt nach sich. Der Täuschungsversuch steht dann in Ihrer Studierendenakte und zieht entspr. Konsequenzen nach sich!**
- Bearbeiten Sie die Aufgabe im Ordner **src/Exam** des Repositories, das ihnen zugewiesen wurde. **Commiten bzw. pushen Sie ihren Code via *git* regelmäßig (ca. alle 10 Minuten) in Ihr zugeteiltes Repository.**
- Schreiben Sie standardkonformen Code, der die Regeln der Veranstaltung für ordentlichen Code und die Vorgaben durch die Seitenklassen berücksichtigt. Verwenden Sie Exceptions zur Behandlung von ungewöhnlichen Fehlern. Entwickeln Sie eine barrierefreie sichere Anwendung. Hinweis: In den PHP-Seitenklassen ist die strenge Typüberprüfung aktiviert. Diese Überprüfung darf nicht abgeschaltet oder umgangen werden.
- Wenden Sie sich bei Fragen an die Prüfungsaufsicht im per Email kommunizierten BBB-Raum.
  Bitte immer nur max. 1 Studierender gleichzeitig im jeweiligen Breakout-Raum!
- Sie können sowohl in Deutsch als auch in Englisch antworten (You may also answer in English).
- Fügen Sie **keine weiteren Dateien** hinzu (diese werden nicht bewertet) und **benennen** Sie die **vorgegebenen Dateien nicht um**; andernfalls werden diese nicht für die Bewertung berücksichtigt!

Wichtig! Schreiben Sie zuerst den Code und versuchen Sie erst am Schluss den Code zum Laufen zu bringen!

Vorbereitung

In dieser Klausur sollen Sie eine Seite entwerfen, die Lehrenden die Möglichkeit eröffnet, in Echtzeit Feedback von Studierenden zur Durchführung einer Lehrveranstaltung zu erhalten. Hierzu gibt es eine Reihe von vordefinierten Reglern (siehe Tabelle "Regler"), über die eine Beurteilung bzw. Rückmeldung zu bestimmten Aspekten (Arbeitsatmosphäre, Konzentration, Vorlesungstempo etc.) abgegeben werden kann. Diese Rückmeldungen werden in der Datenbanktabelle "Bewertung" gespeichert und können anschließend abgerufen und weiterverarbeitet werden.

In Abb. 1. ist ein Mock-up der zu implementierenden Web-Anwendung zur Verdeutlichung dargestellt.

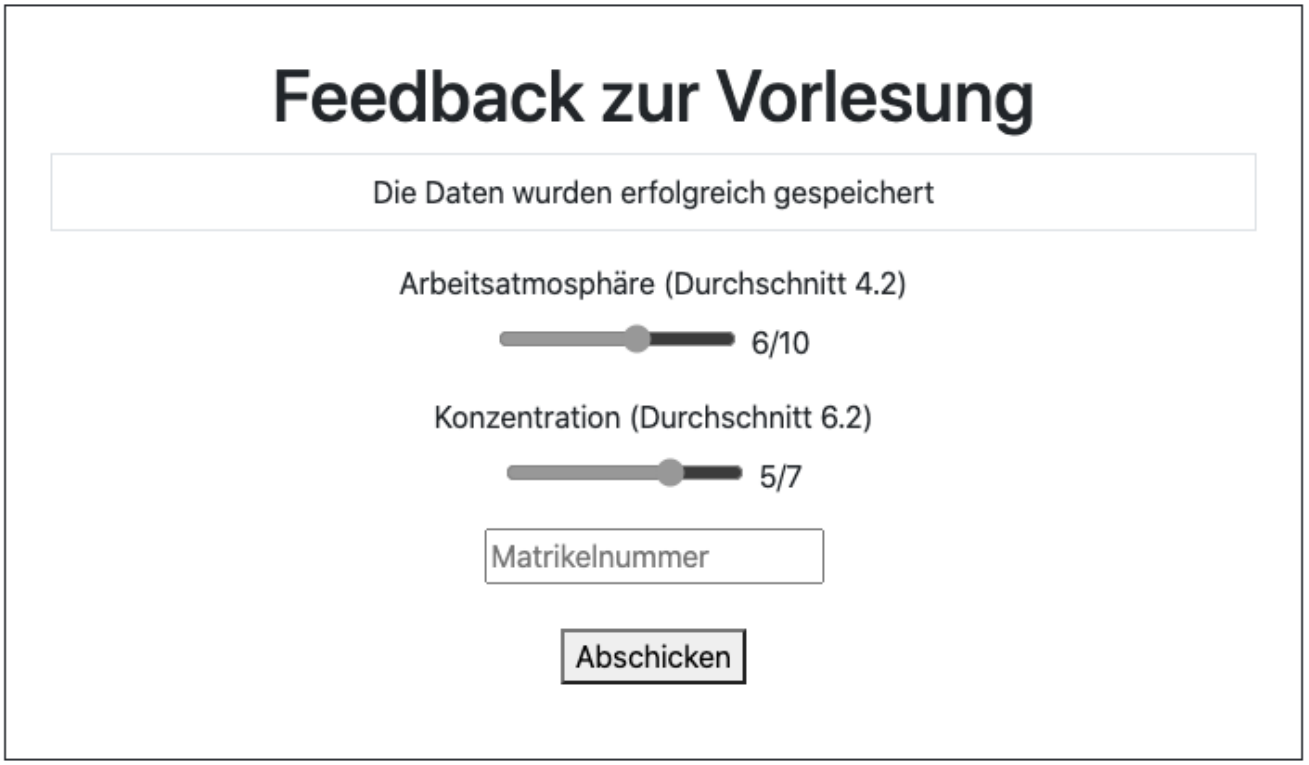


Abbildung 1: Feedbackseite

Die Seite hat eine Hauptüberschrift, darunter einen Bereich für Systemmeldungen an die BenutzerInnen. Anschließend einen Bereich, in dem die Regler dargestellt werden sowie abschließend ein Textfeld für die Matrikelnummer und einen Absende-Button.

Die Regler zur Rückmeldung der verschiedenen Aspekte sollen dynamisch aus der Datenbank (Tabelle "Regler") generiert werden.

Neben der Beschriftung, des jeweiligen Reglers soll der Durchschnittswert aller abgegebenen Bewertungen angezeigt werden (dies jedoch erst in Aufgabe 3).

Wird ein Feedback abgeschickt, erscheint eine Bestätigungsnachricht unterhalb der Hauptüberschrift und die Regler werden auf den Wert vom letzten Feedback gesetzt. Hinweis: Die Seite sendet die Daten also an sich selbst.

Nach dem Festlegen des jeweiligen Wertes müssen die Studierenden im Feld "Matrikelnummer" noch selbige eintragen um sich für eine Rückmeldung zu legitimieren, d.h., durch den Abgleich der eingegebenen Matrikelnummer mit denen in der Tabelle "Student" wird festgelegt, ob ein Studierender eine Rückmeldung zur LVA geben darf (Hinweis: es dürfen nur registrierte Studierende (siehe Tabelle "Student") eine Beurteilung abgeben).

Eine Rückmeldung besteht aus den beiden Regler-Werten sowie der Matrikelnummer.

Die Überprüfung und Verarbeitung dieser Werte übernimmt die Seite selbst (siehe Aufgabe 1e).

Datenbankschema

Die Tabelle: Regler enthält eine Reihe vordefinierter Regler (=Beurteilungsaspekte) und zugehöriger Informationen, die auf einer Seite eingebunden werden können. Jeder Regler besitzt eine Beschriftung, welche den jeweiligen Beurteilungsaspekt benennt, Min.- und Max.-Werte sowie konkrete Bezeichner für den Min.- und Max.-Wert.

Die Tabelle: Student dient dazu abzugleichen, ob es sich bei einer empfangenen Rückmeldung um einen gültigen, d.h., für die LVA registrierten Studierenden handelt.

In der Tabelle: Bewertung werden alle abgegebenen Bewertungen von den in der Tabelle "Student" enthaltenen Studierenden abgespeichert. Um Doppeleintragungen oder Meldungen von unbekannten TeilnehmerInnen zu unterbinden werden dort neben der Regler-ID (d.h., der zu bewertende Aspekt), dem abgegebenen Wert auch die Matrikelnummer gespeichert.

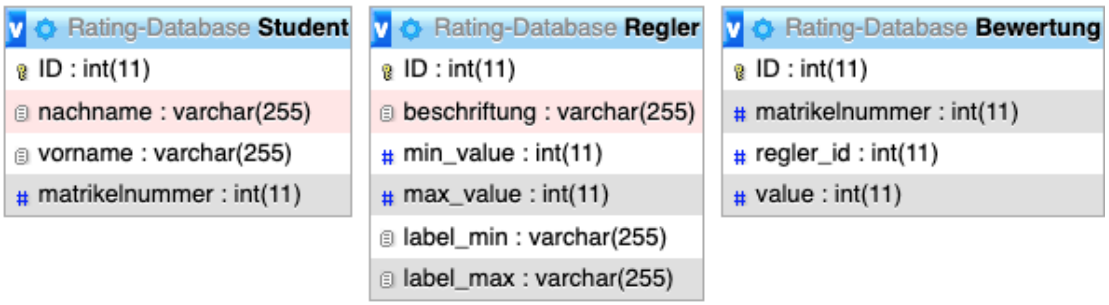


Abbildung 2: Datenbankstruktur

Vorgegebene Werte

In den folgenden Abbildungen 3, 4 und 5 sehen Sie die Inhalte der beschriebenen Tabellen.

Regler

ID	beschriftung	min_value	max_value	label_min	label_max
1	Konzentration	0	7	sehr müde	hellwach
2	Raumtemperatur	0	10	Zu kalt	Zu heiß
3	Vortragslautstärke	0	5	Zu leise	Zu laut
4	Arbeitsatmosphäre	0	10	sehr schlecht	sehr gut

Abbildung 3: Daten der Tabelle "Regler"

Student

ID	nachname	vorname	matrikelnummer
1	Reyes	Hugo	481516
2	Shephard	Jack	548547
3	Locke	Jonathan	263740

Abbildung 4: Testdaten der Tabelle "Student"

Bewertung

ID	matrikelnummer	regler_id	value
1	481516	1	2
2	481516	2	6
3	548547	1	3
4	548547	3	5

Abbildung 5: Testdaten der Tabelle "Bewertung"

Aufgabe 1: PHP **(24 Punkte)**

Implementieren Sie die Feedbackseite (Abb. 1) in der vorgegebenen **Exam.php**-Datei. Verwenden Sie sinnvoll die dazu vorgegebenen Klassenmethoden getViewData(), generateView(), processReceivedData() sowie die Klasse: Page. Die Aufgabe soll ausschließlich mit standardkonformen und barrierefreien HTML und PHP gelöst werden.

Gehen sie wie folgt vor:

Aufgabe 1a:
Überlegen Sie sich ein geeignetes HTML-Element, in welches Sie den gesamten Inhalt (siehe Abb. 1) der Seite packen wollen. Geben Sie der Seite außerdem eine Hauptüberschrift und einen Seitentitel.

Aufgabe 1b:
Generieren Sie zwei Regler. Benutzen Sie dazu die Daten mit der ID 1 und 2 aus der Datenbanktabelle "Regler".

- Erzeugen Sie für jeden Regler eine Beschriftung. Diese soll in ein eindeutig für den Regler vorgesehenes HTML Element.
- Implementieren Sie für jeden Regler weitere sinnvolle Attribute, welche Sie im späteren Verlauf noch benötigen könnten.

Technischer Hinweis für Aufgabe 1b:
Verwenden sich für die Regler folgendes HTML Element: `<input type="range" min="x" max="x" step="1">`

Aufgabe 1c:
Sehen Sie neben jedem Regler ein passendes HTML-Element vor, in das in einer späteren Aufgabe der aktuelle Zahlenwert des Reglers mittels JavaScript geschrieben werden kann. Diese Elemente können Sie mit frei wählbaren Attributen versehen, die Ihnen ggf. in späteren Aufgaben die Verarbeitung erleichtern.

Aufgabe 1d:
Erstellen Sie ein Eingabefeld für die Matrikelnummer. Dieses Eingabefeld hat als Platzhalter das Wort "Matrikelnummer" und ist ein Pflichtfeld d.h. es dürfen keine Daten abgeschickt werden bevor es nicht befüllt ist.

Aufgabe 1e:
Erweitern Sie ihren bisherigen Code so, dass Sie alle benötigten Daten abschicken können und diese sicher in der Datenbanktabelle "Bewertung" gespeichert werden. Eine Bewertung besteht aus: Regler-Id, Matrikelnummer und dem Regler-Wert.

Wurden die Daten verschickt soll folgendes passieren:

- Prüfen Sie als Erstes ob die mitgeschickte Matrikelnummer in der Datenbanktabelle "Student" enthalten ist. Ist das der Fall kann eine neue Bewertung angelegt werden. Existiert bereits eine Bewertung mit der gleichen Matrikelnummer, soll diese stattdessen aktualisiert werden.
- Realisieren Sie die Aufgabe mit dem **PRG-Pattern**.

Aufgabe 1f:
Erzeugen Sie unterhalb der Hauptüberschrift einen Textausgabebereich. In diesem soll eine generierte Rückmeldung geschrieben werden, je nachdem ob die Eintragung erfolgreich war oder ein Fehler vorliegt (bspw. ob die Matrikelnummer nicht gefunden werden konnte). Jeder Regler sollte außerdem nun auf den zuletzt abgeschickten Werten stehen.

Tipps für Aufgabe 1f:
- Überlegen Sie sich, wo Sie die generierten Nachrichten und erhaltenen Werte seitenübergreifend zwischenspeichern können!

Aufgabe 2: JavaScript **(12 Punkte)**

Standardmäßig zeigt Exam.php (=die Feedbackseite) nur jeweils den Schieberegler, nicht jedoch den aktuell gewählten Wert als Ganzzahl an.

Erweitern Sie deshalb die Datei Exam.js derart, dass nun mittels JavaScript der aktuell gewählte Wert eines Reglers rechts neben dem Regler angezeigt wird und sich der Wert bei jeder Änderung automatisch aktualisiert.
Ist der Min.- oder Max.-Wert eines Reglers erreicht, so soll der in der Tabelle "Regler" hinterlegte zugehörige Text neben dem aktuellen Zahlenwert angezeigt werden.

Hinweise:

- Die beschriebene Funktionalität soll ausschließlich mittels JavaScript realisiert werden Es genügt, wenn Sie für die Aktualisierung des Zahlenwertes nur das Input-Event berücksichtigen
- Lagern Sie den Code zur Aktualisierung der Zahlenwerte in die Funktion updateValue(...) aus
- event.target liefert den DOM-Knoten auf dem das Event erzeugt bzw. "gecaptured" wurde.
- Bedenken Sie, dass es auch andere Regler in der DB gibt, d.h., Ihre Lösung soll prinzipiell mit allen anderen Reglern funktionieren können. Die volle Punktzahl erhalten Sie nur für eine derartige generalisierte Lösung.
- Überlegen Sie sich ein geeignetes Vorgehen, wie Sie die Beschriftungen für die Min.- & Max.-Werte aus der DB im DOM verfügbar machen. Beschreiben Sie Ihr Vorgehen kurz in einem Kommentar in der Funktion updateValue(...).
- Wichtig: Der JavaScript-Code soll erst aktiv werden, wenn die Seite vollständig vom Browser geladen und verarbeitet wurde.

Aufgabe 3: AJAX **(16 Punkte)**

Ergänzend zur Anzeige des aktuellen Reglerwertes soll auch der Durchschnittswert aller eingegangenen Werte für einen Regler ausgegeben werden. Diese Werte sind in der Tabelle "Bewertung" gespeichert (siehe Aufgabe 1) und werden über die zugelieferte Datei RatingsAPI.php[1] als serialisierte JSON-Zeichenkette zurückgeliefert.

```
[ {"Regler-ID": "1", "Bewertungen": [5, 4, 2, 1, 7, 3, 4, 2]},
  {"Regler-ID": "2", "Bewertungen": [2, 1, 1, 4, 1, 1, 2, 2]}
]
```

Diese Zeichenkette soll mittels AJAX zyklisch abgerufen und verarbeitet werden. Hierbei sollen die Einzelwerte pro Sensor addiert und ein Durchschnittswert berechnet werden. Der Durchschnittswert soll anschließend neben der Beschriftung des jeweiligen Reglers in dem folgenden Format erscheinen[2]:

Arbeitsatmosphäre (Durchschnitt: 4.5)

Erweitern Sie die Datei Exam.js hierfür um folgende Funktionalität:

- Implementieren Sie eine Funktion fetchRatings(), welche die notwendigen Aufrufe asynchron ausführt.
- Die Durchschnittswerte sollen alle 2 Sekunden aktualisiert werden.
- Schreiben Sie eine asynchrone Funktion calculateAverage(ratings), welche alle Einzelwerte eines Reglers als Argument übergeben bekommt und daraufhin den Durchschnittswert berechnet..
- Führen Sie alle Aktualisierungen des DOM in der Funktion updateHTML(...) aus (Selektion des entsprechenden Reglers + Ergänzung der Beschriftung).
- Implementieren Sie diese Aufgabe analog dem empfohlenen Vorgehen für die Entwicklung asynchroner Aufrufe/Funktionen für die volle Punktzahl[3].
    - Benutzen Sie async + await
    - Sichern Sie kritische Codestellen
    - Verwenden Sie die fetch-API

Tipps:

- Achten Sie auf das korrekte Zusammenspiel der jeweiligen Funktionen
- Die Regler-ID im JSON entspricht der Regler-ID in der Datenbank (siehe Tabelle "Regler")
- **Definieren Sie ein Konstante als reference value mit Namen fetchRatings und weisen sie diesem eine anonyme asynchrone Funktionsdefinition zu, in der Sie alle notwendigen Schritte implementieren.**

[1] Hinweis: Die Datei "RatingsAPI.php" liefert nur einen Test-Datensatz nicht jedoch die tatsächlich in der Tabelle hinterlegten Werte.
[2] Beispiel für den Regler "Arbeitsatmosphäre" (siehe Tabelle "Regler" und auch Abb. 1)
[3] Sie können die Aufgabe auch losgelöst von den o.g. Empfehlungen implementieren, bekommen dann allerdings nicht die volle Punktzahl.

Aufgabe 4: Wissensfragen **(22 Punkte)**
Bitte beantworten Sie alle Fragen in der Datei Exam.txt

Aufgabe 4a: (2 Punkte)
Zeigen Sie an einem einfachen selbst gewählten Beispiel, wann ein Objekt in JavaScript zu einem Reference Type wird; was ist hierzu notwendig?

Aufgabe 4b: (2 Punkte)
Nehmen Sie Stellung zu folgenden Thesen:
- "Eine Funktion in JavaScript ist immer ein reference value"

Aufgabe 4c: (3 Punkte)
Gegeben ist folgender Codeausschnitt:

```
function Book(title, author) {
   this.title = title;
   this.author = author;

   this.print = function(aspect) {
       console.log(this[aspect]);
   }
}
let book1 = new Book("Die verlorene Ehre der Katharina Blum", "Heinrich Böll");
```

Die Eigenschaft printTitle() soll allen aktuellen sowie allen zukünftigen Objekten vom Typ "Book" zur Verfügung stehen. Zeigen Sie auf, mit welchem Sprachmittel / Konzept sich dies in JavaScript umsetzen lässt.

Aufgabe 4d: (5 Punkte)
Implementieren Sie die in Aufgabe 3 beschriebene Funktion calculateAverage(ratings) als Closure.

Aufgabe 4e: (8 Punkte)
Legen Sie dar, warum es für ein Closure unerheblich ist, ob es als Konstruktorfunktion oder durch einen "normalen" Funktionsaufruf instanziiert wird.

Aufgabe 4f: (2 Punkte)
Bei einem Onlinekatalog soll sich die Auswahl der anzuzeigenden Artikel individuell festlegen lassen. Auf Grundlage welcher HTTP-Kommunikationsmethode realisieren Sie dies vorzugsweise? Begründen Sie Ihre Antwort kurz.

Aufgabe 5: CSS **(6 Punkte)**

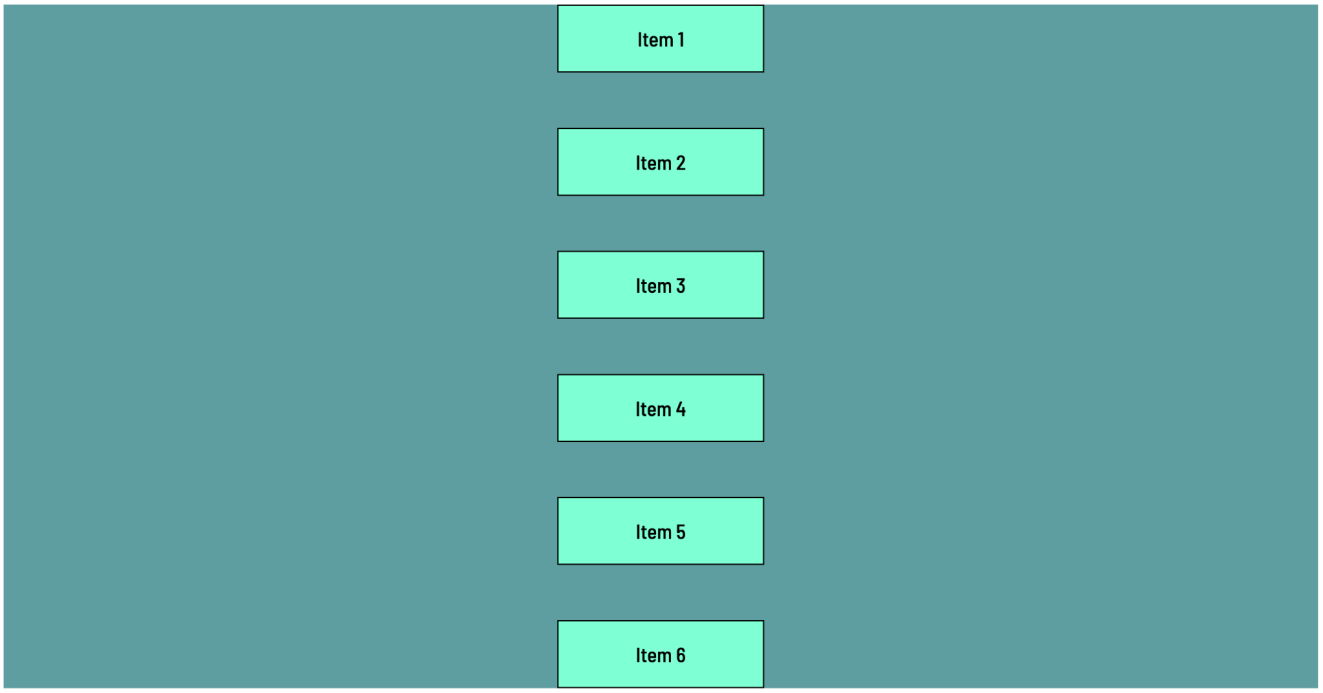
Gegeben sei folgende HTML-Datei:

```
<!doctype html>
<html>
<head>
    <title>EWA-Klausur WS 2021/2022: CSS-Aufgabe</title>
    <meta charset="utf-8" />
</head>
<body>
    <div id="container">
        <div class="item">Item 1</div>
        <div class="item">Item 2</div>
        <div class="item">Item 3</div>
        <div class="item">Item 4</div>
        <div class="item">Item 5</div>
        <div class="item">Item 6</div>
    </div>
</body>
</html>
```

Lösen Sie nachfolgend gestellte Aufgaben mittels dem Flexbox-Modul[4]. Tragen Sie hierzu die entsprechenden CSS-Stilregeln und Selektoren in die Daten "Exam.css" ein.

A) Die Items sollen grundsätzlich wie in nachstehender Abbildung platziert werden:

B) Unterhalb einer Bildschirmbreite von 700 Pixeln soll die Anordnung wie dargestellt erfolgen:



C) Fügen Sie noch ein Stilregel bzgl der Schriftgestaltung in den Items hinzu; diese sollen in einer Serifen-Schrift, vorzugsweise in "Cambria" dargestellt werden.

[4] Bei Aufgabe 5 geht es ausschließlich um die Anordnung der Items mittels der Flexbox; die in den Screenshots gezeigte Farbgestaltung etc. brauchen Sie nicht umzusetzen!

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.

Hinweise für den Notfall

Falls Sie Probleme mit der Netzwerkverbindung haben...

- Ruhe bewahren und erst einmal weiterarbeiten. Oft lösen sich die Probleme nach ein paar Minuten.
- Das Handy für den Netzwerkzugang nutzen (USB-Tethering)
- Bescheid geben / Kontakt aufnehmen über den 2. Kanal (Zoom evtl. mit dem Handy). Die Zugangsdaten erfahren Sie vor der Prüfung per Email.
- Nur wenn die Probleme bis zur Abgabe bestehen, sollten Sie das Endergebnis als ZIP-Datei per Email an Prof. Dr. Stefan Zander senden. In diesem Fall gilt der Zeitstempel des Emailproviders als Abgabezeit.