Projektspezifikation „Appointment Finder“

I. Allgemein

a. Im Rahmen der Lehrveranstaltung Web Scripting muss ein Web-Projekt umgesetzt werden.
b. Verwenden Sie für die Umsetzung Teile Ihrer bereits abgegebenen Übungen wieder.
c. Die Bearbeitung erfolgt in 2er Teams. Einzelarbeiten sind nur in Ausnahmefällen zulässig.
    i. **Blau markierte Aufgaben sind nur von 2er Teams zu bearbeiten, Einzelarbeiten lassen diese aus! (Punkte für blau markierte Aufgaben werden nicht zu der Gesamtpunktezahl für Einzelprojekte addiert)**
d. Das Abnahmegespräch erfolgt dann im Team (bzw. im Ausnahmefall einzeln). Jedes Teammitglied muss das Projekt (Funktionsumfang und Code) kennen und Fragen dazu beantworten können.
    i. Das Abnahmegespräch orientiert sich an einer Bewertungsmatrix. Diese steht vorab zur Verfügung und dient u.a. der Darstellung der Gewichtung der einzelnen Aufgaben sowie zur Kontrolle der Vollständigkeit.
e. Für manche in grün markierte Tasks, gibt es Extrapunkte zu erreichen. Diese Aufgaben sind optional.
f. Die Angabe lässt gewisse Gestaltungs- und Implementierungsfreiheiten zu. Folgende **Basis-Features** müssen abgebildet werden:
    i. Darstellung aller Appointments in einer Liste (= Startseite)
    ii. Detailansicht eines Appointments inkl. aller Terminoptionen und bisherigen Votings (inkl. Kommentaren)
    iii. Nutzer*innen können in einer Detailansicht für einen/mehrere Termine „voten“
    iv. Neu-Anlegen eines Appointments (z.B.: Titel, Ort, Info, Dauer) inklusive einer Liste von Terminvorschlägen
g. Die Umsetzung des Projekts wird **sämtliche Inhalte** dieser LV und ausgewählte Inhalte der LV Webtechnologien des 1. Semesters abdecken:
    i. Frontend: HTML5, CSS3, Bootstrap

ii. JavaScript, jQuery, AJAX
iii. TypeScript
iv. PHP (für REST-Backend und zur Datenbankanbindung)
v. MySQLi + Prepared Statements im Zuge der Datenbankanbindung

h. Achten Sie auf eine **gute User Experience** innerhalb der Applikation (keine Whitepages, keine Fehlerzustände, guter Gesamteindruck, …)

II. Projektstruktur und Grundlayout

a. Erstellen Sie eine HTML-Datei für das Frontend (index.html – keine php-Datei), welche das **Grundlayout** enthält und die Hauptbereiche des „Appointment Finders“ abdeckt, sowie CSS- und JavaScript-Files einbindet. Alle geforderten Bereiche befinden sich auf dieser index.html-Seite. Seiteninhalte sollen dynamisch nachgeladen werden, ohne einen Refresh der gesamten Webseite notwendig zu machen (Single Page Application). Ein Button „New Appointment“ leitet den User im späteren Projektverlauf zu einer Eingabemaske.
b. Die Projektstruktur soll die einzelnen **Komponenten** klar **unterteilen** (siehe Beispielstruktur).

Beispielstruktur:

```
[frontend]
    index.html
    …
    [css]
        mystyle.css
        …
    [js]
        myscript.js
        …

[backend]
    serviceHandler.php
    [businesslogic]
        simpleLogic.php
        …
    [db]
        db.php
        dataHandler.php
    [models]
        appointment.php
```

c. Zur Entwicklung des Backends verwenden Sie die in Block 4 verwendeten Code-Bausteine als Grundlage. Legen Sie zusätzlich eine DB-Service-Klasse an, welche **zentral** die Zugriffe auf die DB implementiert.
    i. Achten Sie bei der Entwicklung darauf, dass Sicherheitslücken beim Zugriff auf die Datenbank möglichst geschlossen werden.
d. Legen Sie für dieses Projekt eine neue **Datenbank** an.

III. Frameworks und Libraries

a. Es bleibt Ihnen überlassen, ob Sie für die Implementierung diverse Frameworks oder Bibliotheken verwenden, wie z.B. Bootstrap, YAML, …
b. Die Verwendung von Fertiglösungen (wie CMS o.ä.) ist nicht erlaubt!

IV. Abgabemodalitäten

a. Das Projekt ist **vollständig** als zip/rar File in Moodle hochzuladen (maximale Dateigröße 20MB), inklusive einer Kopie der MySQL-Datenbank (sql-Anweisungen) und einem Hinweis auf die Teammitglieder im Dateinamen, z.B.:
    i. **WEB-Projekt-SS2021 _Nachname_Nachname**.zip
        - Halten Sie sich unbedingt an die Nomenklatur, um die korrekte Zuweisung und somit Bewertung von gemeinsam abgegeben Projekten sicherzustellen!
        - Verkleinern Sie Bilddateien ggf., um die Größe des Projekts für den Upload zu reduzieren.
b. **Kommentieren** Sie komplexere Abläufe/Funktionen im Code.
c. Falls weitere Konfigurationsschritte zur Inbetriebnahme der Applikation nötig sind, legen Sie bitte eine **readme.txt**-Datei bei.

Grundidee

Nutzer*innen der Website haben die Möglichkeit Terminvorschläge mittels Checkboxen auszuwählen und ihren Namen für diese Terminauswahl (Voting) zu hinterlassen (Sie kennen bestimmt vergleichbare Webseiten zur Terminfindung für mehrere Personen, wie z.B. doodle.com).

Dabei gilt es, ein Backend mit Webservice (auf php-Basis) und ein Web-Frontend (auf HTML / JavaScript / TypeScript-Basis) zu entwickeln.

Die gesamte Website muss **responsive** sein und somit für Smartphones, Tablets und Desktop-Rechner verwendbar sein.

Für eine außergewöhnlich gute **Usability** und Grundzüge der **Accessibility** werden Zusatzpunkte gewährt.

User

Es gibt nur einen Usertyp: Nutzer*innen der Webseite. Ein Login auf der Webseite ist der Einfachheit halber nicht zwingend notwendig.

Nutzer*innen haben folgende Möglichkeiten:

1. Darstellung aller Appointments in einer Liste (Startseite)
2. Detailansicht eines Appointments (durch Anklicken eines Appointments aus der Liste)
    a. Eingabe des eigenen Namens
    b. Anzeigen von verfügbaren Terminoptionen
    c. Auswahl beliebiger Terminoptionen (Voting)
    d. Kommentar hinterlassen
    e. Achtung: Appointments haben ein Ablaufdatum. Danach ist ein Voting, also die Auswahl von Terminoptionen nicht mehr möglich
3. **Neues Appointment anlegen**
4. **Appointments löschen**

Aufbau der Datenbank

Datenbankmodell:

Überlegen Sie sich ein **geeignetes Datenbankmodell** und legen Sie eine neue Datenbank an. Hinweise: Sie benötigen mehrere Tabellen. Achten Sie darauf, dass mindestens folgende Daten gespeichert werden können:

- Appointments (mit Titel, Ort, Datum, Ablaufdatum des Votings etc.)
- auswählbare Termine
- Usernamen mit ausgewählten Terminen und hinterlassenen Kommentaren

Datenbankzugriff mittels PHP:

Legen Sie einen neuen User mit folgenden Details in phpMyAdmin an:

- username: bif2webscriptinguser
- password: bif2021
- Rechte: localhost, datenbankspezifisch, ALL PRIVILEGES

Der Datenbankzugriff hat im Projekt über **mySQLi** zu erfolgen (alternativ ist auch PDO möglich).

Die Zugangsdaten sollen in einer **config-Datei** gespeichert werden, damit sie an zentraler Stelle leicht zu ändern sind.

Die Kommunikation zwischen Frontend und Datenbank MUSS über das **Webservice** realisiert werden! Stellen Sie sich vor, es sollen in Zukunft – neben einer Web-Oberfläche – noch weitere Frontends (Windows-App, Android-App, etc.) entwickelt werden, welche die Requests an denselben serviceHandler schicken, um (von unterschiedlichen Frontends) an einer gemeinsamen Terminfindung teilnehmen zu können. Für dieses Projekt ist nur ein Web-Frontend gefordert.

Beschreibung der einzelnen Komponenten

1) Darstellung aller Appointments in einer Liste

Nutzer*innen der Webseite wird auf der Startseite eine Liste aller Appointments dargestellt. Appointments können abgelaufen sein und in der Liste als solche gekennzeichnet sein.

2) Detailansicht eines Appointments

Nutzer*innen können auf die einzelnen Appointments (welche auf der Startseite dargestellt werden) klicken, um eine Detailansicht zu erhalten. Darin sind alle Terminoptionen dieses Appointments sowie alle bisherigen Votings und Kommentare einsehbar. In dieser Detailansicht können Nutzer*innen auch für ein/mehrere Terminoptionen „voten“ (sofern das Ablaufdatum noch nicht erreicht ist).

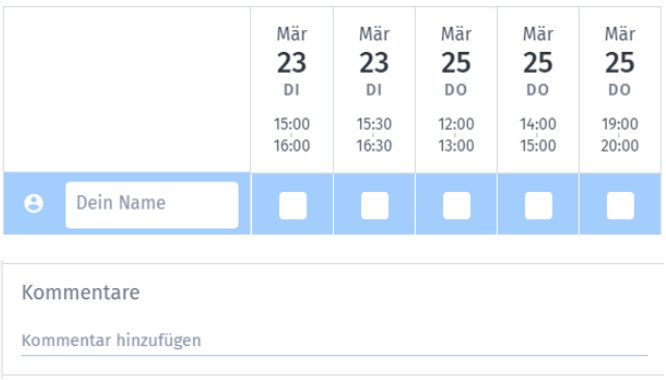
a. Eingabe des eigenen Namens

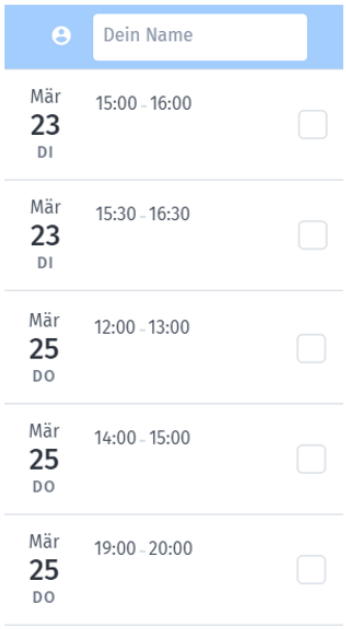
Beim Voten für verfügbare Terminoptionen, muss der Name des Users erfasst werden können, damit später nachvollzogen werden kann, welcher User welche Terminoptionen ausgewählt hat.

b. Anzeigen von verfügbaren Terminoptionen

Über das Webservice werden die verfügbaren Terminoptionen abgerufen und im Frontend angezeigt.

Mögliche Darstellungsformen (zum einfacheren Verständnis):





(Screenshots von https://doodle.com/)

Verwenden Sie nicht zu viel Zeit für das Styling, konzentrieren Sie sich auf die Funktionalitäten!

c. Auswahl beliebiger Terminoptionen

Für das Voting von Terminoptionen können Sie z.B. Checkboxen neben/unter jeder Terminoption verwenden. Aktivierte bzw. nicht aktivierte Checkboxen sollen in der Tabelle der Datenbank entsprechend gespeichert werden. Dies kann z.B. mit 0/1 oder mit leeren/nichtleeren Einträgen erfolgen.

Optionales Feature: Terminoptionen können nachträglich (nicht) veränderbar sein.

d. Kommentar hinterlassen

Jeder User hat die Möglichkeit Kommentare zu hinterlassen, welche gemeinsam mit seinen ausgewählten Terminoptionen in der Datenbank gespeichert werden sollen. Kommentare sollen optional sein.

Nachdem alle Eingaben getätigt wurden, sollen diese über einen Button an eine bestimmte Funktion des serviceHandlers übergeben werden, welche die Eintragung in die Datenbank übernimmt.

3) Neues Appointment anlegen (nur für 2er Teams)

Ergänzen Sie das Frontend um einen Button „New Appointment“.
Dieser soll dem User (in der Single Page Application) eine Eingabebaske zur Verfügung stellen. In dieser Eingabemaske sollen alle notwendigen Elemente erfasst werden, um Terminoptionen für eine neue Terminfindung (über den serviceHandler) anzulegen. z.B. Titel, Ort, Beschreibung, Dauer, Terminoptionen. Hinweis: Achten Sie darauf, dass die Datenbank entsprechend angepasst wird.

Einzelarbeiten verwenden hard-codierte Daten in der Datenbank, um bereits eingetragene Appointments auszulesen!

4) Appointments löschen

Bestehende Appointments sollen (über den serviceHandler) gelöscht werden können.

Optionaler Working Plan zum Einstieg in die Projektentwicklung (Vorschlag)

Um Ihnen die Orientierung beim Umsetzen der Anforderungen zu erleichtern, wird folgendes Vorgehen vorgeschlagen:

Starten Sie mit der Entwicklung des Backends. Kopieren Sie dazu die Code-Vorlage aus Block 4 in ein eigenes Projektverzeichnis und überlegen Sie, welche Änderungen durchzuführen sind (statt Personen werden in diesem Projekt Terminvorschläge benötigt). Passen Sie dann die Klassen und Funktionen schrittweise entsprechend an.

person.php wird z.B. zu appointment.php mit entsprechenden Parametern. Die Funktion getDemoData() *in der* DataHandler*-Klasse kann für die ersten Entwicklungsschritte im Frontend dahingehend adaptiert werden, dass – dem Projekt entsprechend – sinnvolle Demo-Daten (Terminvorschläge) verwendet werden.*

> *Im späteren Projektverlauf sollen statt hardcodierten Demo-Daten, Daten aus einer Datenbank verwendet werden. Diese Klasse wird in diesem Zuge dann auch um eine Funktion erweitert, mit welcher neue Datenbankeinträge durchgeführt werden können.*

Für die ersten Entwicklungsschritte können bei Bedarf Demo-Daten verwendet werden.

Passen Sie darauf aufbauend die weiteren Funktionen in den Klassen DataHandler und SimpleLogic *sowie im* serviceHandler *an, damit Requests von verschiedenen Frontends durchgeführt werden können. Stellen Sie sich vor, es sollen in Zukunft – neben einer Web-Oberfläche – noch weitere Frontends (Windows-App, Android-App, etc.) entwickelt werden, welche die Requests an denselben serviceHandler schicken, um (von unterschiedlichen Frontends) an einer gemeinsamen Terminfindung teilnehmen zu können. Für dieses Projekt ist nur ein Web-Frontend gefordert.*

Nachdem eine erste Version des Backends fertig ist, starten Sie mit der Entwicklung des Frontends, in welchem die verfügbaren Terminvorschläge (aus der Funktion getDemoData()*) abgerufen werden können. Sie können als Code-*

Vorlage den simpleJsonClient *(html & js File) aus Einheit 4 heranziehen und darauf aufbauen.*

Erstellen Sie die notwendigen Elemente in HTML und JavaScript (z.B. via jQuery & TypeScript), um die verfügbaren Terminoptionen im Frontend anzuzeigen.

*Nutzer*innen sollen ihren eigenen Namen angeben und ihre präferierten Terminvorschläge (z.B. via Checkbox) auswählen können (= voten).*

Beginnen Sie was das Frontend betrifft mit einem Bootstrap Basislayout, welches responsive ist.

1. *Überlegen Sie, welche Elemente in welcher Form dargestellt werden sollen.*
    a. *Setzen Sie schon vorgefertigte Bootstrap-Komponenten ein*
2. *Ergänzen Sie das individuelle Design (aber halten Sie sich nicht zu lange damit auf!)*
    a. *Eigene CSS-Datei*
    b. *…*
3. *Erstellen Sie alle notwendigen Elemente*
    a. *input-Feld für Name der Nutzerin / des Nutzers*
    b. *Checkboxes auf Basis der Anzahl der hinterlegten Terminvorschläge*
    c. *Button*
4. *Beginnen Sie schon jetzt mit dem Kommentieren Ihres Codes und halten Sie offene/zukünftige Punkte fest*
5. *Nehmen Sie mittels JavaScript und AJAX die Demo-Daten vom* serviceHandler *entgegen und zeigen Sie diese (z.B. mittels Schleife und jQuery) im Frontend an.*

Sobald das Frontend mit den Demo-Daten funktioniert, können Sie zu den nächsten Schritten übergehen, um den Code um Datenbankstatements zu erweitern.

6. *Überlegen Sie sich Möglichkeiten, um Daten, die im Frontend eingegeben werden, in der Datenbank zu speichern. Verwenden Sie dazu das Webservice! (*serviceHandler*,* SimpleLogic*,* DataHandler*, etc.)*

   a. *Entwerfen Sie ein erstes DB-Modell, das alle benötigten Daten aufnehmen und abbilden kann. Erweitern/Adaptieren Sie dieses ggf. im Laufe der Entwicklung.*
7. *Als ersten Schritt könnten Sie die Demo-Daten in einer Datenbank abbilden und den Code zur Abfrage der Daten über die* DataHandler-*Klasse entwickeln.*
8. *Fügen Sie die weiteren Funktionalitäten im Anschluss schrittweise hinzu:*
   a. *Daten vom Frontend dem* serviceHandler *übergeben und entsprechende Funktionen anlegen.*
   b. *Vergessen Sie nicht, den Code zu kommentieren und zu testen!*