Fahrtenprojekt

Inhaltsverzeichnis

---

1. Einleitung

Beschreibung

Dieses Fahrtenbuch hilft dabei, Fahrten zu speichern. Im Formular können folgende Daten eingegeben werden:

- Name
- Uhrzeit
- Datum
- Startkilometer
- Endkilometer

Die Daten werden in einer **XML-Datei** gespeichert und in einer Tabelle angezeigt.

Projektziele

Das Ziel des Projekts ist es, eine Webanwendung zu entwickeln, mit der Fahrten dokumentiert und angezeigt werden können. So kann die Person, die das Fahrtenbuch nutzt, ihre gefahrenen Strecken einfach eintragen und nachverfolgen.

Verwendete Technologien

- **Frontend**: HTML, CSS, JavaScript
- **Backend**: PHP
- **Datenverarbeitung**: XML

Projektmanagement

- **Methode**: Es wurde Methode scrum benutzt,
- Zur Verfolgung wurde **Trello-Board** benutzt Link zum Board
- für Version Control haben uns für Git-Hub entschiden **Repository**:

```
git clone https://github.com/cheechay/Fahrtenprojekt
```

---

2 Teammitglieder und Aufgabenverteilung

Projekt-Owner

- **Oliver Metzger**

Sprint 1

Shella Mae Friedlein

- **Rolle**: Entwickler/in
- **Aufgaben**:
    - Erstellung des Eingabeformulars
    - Design des Formulars

Zhazmira Borubaeva

- **Aufgaben**:
    - Scrum Master
    - Entwicklung des Ausgabeformulars
    - Projektdokumentation

Priyanka KC

- **Aufgaben**:
    - XML-Datenverarbeitung mit PHP
    - Korrektur falscher Daten in Auflistung

---

3 Anforderungen an das Projekt

1. **Validierung**:
    - Das Eingabeformular muss validiert werden, um sicherzustellen, dass alle Eingaben korrekt und vollständig sind.
2. **Datenverarbeitung**:
    - Die eingegebenen Daten müssen korrekt gespeichert und später fehlerfrei ausgegeben werden.
3. **Ausgabeformat**:
    - Das Ausgabeformular soll die Daten in einem übersichtlichen Tabellenformat darstellen.
4. **Benutzerfreundlichkeit**:
    - Sowohl das Eingabe- als auch das Ausgabeformular müssen übersichtlich und leicht verständlich gestaltet sein.
5. **Einheitenprüfung**:
    - Die eingegebenen Daten müssen mit den richtigen Einheiten (z. B. Kilometer, Datumsformate) versehen und verarbeitet werden.

---

4 Technische Umsetzung

Projektstruktur

1. Eingabeformular - index.html

Das Eingabeformular ermöglicht es den Nutzern, relevante Fahrtenbuchdaten wie Name, Wohnort, Zweck, Datum und Kilometerstände zu erfassen. Das Formular nutzt HTML zur Strukturierung und CSS für das visuelle Design. Es wird durch JavaScript validiert, um sicherzustellen, dass alle Pflichtfelder korrekt ausgefüllt werden, bevor die Daten übermittelt werden.

Wichtige Funktionen:

Eingabefelder: Ermöglichen die Eingabe von Name, Wohnort, Zweck, Datum, Kilometerständen und mehr. Google Maps Verlinkung: Ein eingebautes Google Maps-Icon bietet einen direkten Link zur Kartenanwendung, um den Standort schnell zu finden.

Datenvalidierung: Verhindert die Übermittlung unvollständiger oder fehlerhafter Formulardaten.

Der HTML-Code enthält auch eine benutzerfreundliche Gestaltung mit Feldern für die einfache Dateneingabe. Außerdem sind die Felder name, wohnort, und zweck als verpflichtend **(required)** validiert.

Automatisierte Berechnungen: Eine Funktion zur Berechnung der gefahrenen Kilometer (Differenz zwischen km_start und km_ende) wurde implementiert werden.

```
<html>

<body>

<!-- VERKÜRZTE CODE -->

        <form action="index.php" method="POST">
            <fieldset>
                <legend>Tragen Sie ein</legend>

                <table class="info-section">
                    <tr>
                        <td><label for="name"> Name</label></td>
                        <td colspan="2"><input type="text" name="name"
id="name" size="35" required></td>
                        <td rowspan="3">
                            <a href="https://www.google.com/maps/"
target="_blank"><img src="map.png" alt="google map"
                                    width="100" height="115"></a>
                            <p>Suchen auf Google map </p>
                        </td>

                        <!-- Tabelle: <td> + inputs und labels Tags des
Codes -->
<!--
                         Button "Abgeben", der übernimmt dir Javascript
Funktion testEingabe(), um die daten (km_start und km_ende) zu Valideiren: -
->

                    <div class="btn-submit" style="margin-left:50px;">
                        <button type="submit" onclick="return
testEingabe()"> Abgeben <i class="fa-regular fa-circle-right"></i></button>
                    </div>
            </fieldset>
        </form>
```

```
</body>
</html>
```

2. Datenverarbeitung mit PHP:

Die Verarbeitung und Speicherung der Benutzereingaben in einer XML-Datei. Das wird durch die Verarbeitung des Formulars in PHP realisiert.

2.1. Eingabedaten aus dem Formular

Die Benutzerdaten werden mittels $_POST-Methode an das Backend übermittelt und anschließend mit der Funktion filter_input() validiert.

```
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $name = filter_input(INPUT_POST, 'name', FILTER_DEFAULT);
    $wohnort = filter_input(INPUT_POST, 'wohnort', FILTER_DEFAULT);
    $zweck = filter_input(INPUT_POST, 'zweck', FILTER_DEFAULT);
    $datum = $_POST['datum'];
    $km_start = filter_input(INPUT_POST, 'km_start', FILTER_DEFAULT);
    $km_end = filter_input(INPUT_POST, 'km_end', FILTER_DEFAULT);
    $uhrzeit_von = filter_input(INPUT_POST, 'uhrzeit_von', FILTER_DEFAULT);
    $uhrzeit_bis = filter_input(INPUT_POST, 'uhrzeit_bis', FILTER_DEFAULT);

    // Berechnung der Kilometerdifferenz
    $kmdiff = (int) $km_end - (int) $km_start;
}
```

2.2. Datenvalidierung und Fehlerbehandlung

Der Kilometer-Endwert (km_end) darf nicht kleiner sein als der Kilometer-Startwert (km_start).

Die Endzeit (uhrzeit_bis) darf nicht vor der Startzeit (uhrzeit_von) liegen.

```
if (uhrzeit_von >= uhrzeit_bis && kmStart >= kmEnd) {
    alert('Bitte, Eingaben überprüfen☹  !!');
    return false;
}
```

2.3. Speicherung der Daten in einer XML-Datei

In PHP wird dazu ein SimpleXMLElement-Objekt genutzt, um die Daten in die XML-Struktur einzufügen.

```
$xml = simplexml_load_file('fahrtenbuch.xml'); // Laden der bestehenden XML-Datei
$entry = $xml->addChild('entry');
$entry->addChild('name', $name);
$entry->addChild('wohnort', $wohnort);
usw.
$xml->asXML('fahrtenbuch.xml'); // Speichern der aktualisierten XML-Datei
```

```
<data>
  <entry>
    <distance>15.3</distance>
    <date>2025-01-20</date>
  </entry>
</data>
```

3 Ausgabe-Formular

Datenquelle (XML-Datei):

Alle Einträge des Fahrtenbuchs werden aus einer XML-Datei (fahrtenbuch.xml) geladen. Das ist eine einfache Möglichkeit, um Daten zu speichern.

- Die **$_GET-Anfrage** wird verwendet, um daten zu bekommen.
- Wenn ein Eintrag gelöscht werden soll, wird der entsprechende Eintrag aus der XML-Datei entfernt und die Datei neu gespeichert.
- Das XML-Dokument wird nach dem Löschen neu geladen, um die aktuelle Version anzuzeigen.
- Die foreach-Schleife durchläuft die XML-Daten und gibt für jeden Eintrag die relevanten Informationen in der Tabelle aus
- Fehlerbehandlung.

Das Formular überprüft, ob die XML-Datei korrekt geladen werden kann und gibt eine Meldung aus, wenn keine Fahrten vorhanden sind

```
$xmlFile = 'fahrtenbuch.xml';
$xml = simplexml_load_file($xmlFile) or die("<p>Es sind gerade keine Fahrten
eingetragen</p>");

$id = isset($fahrt['id']) ? $fahrt['id'] : '';
$delete_id = isset($fahrt['id']) ? $fahrt['id'] : '';

$fahrtenArray = [];

foreach ($xml->fahrten as $key => $fahrt) {
    // Überprüfe, ob das 'id'-Attribut der ID entspricht
    if ((string)$fahrt['id'] === (string)$deleteId) {
        // Lösche das Element
        unset($xml->fahrten[$key]);
    }
}
```

5 Verwendete Quellen/Literatur

Quelle für Eingabe Formular:

https://www.w3schools.com/xml/xml_whatis.asp

Quelle für PHP und XML lernen und die Daten ausgeben:

https://www.php.net/manual/en/simplexml.examples-basic.php

Example #3 Getting `<line>`

Run code

```
<?php
include 'example.php';

$movies = new SimpleXMLElement($xmlstr);

echo $movies->movie->{'great-lines'}->line;
?>
```

- > (cmd+click);

Run code

```php
<?php
include 'example.php';

$movies = new SimpleXMLElement($xmlstr);

/* For each <character> node, we echo a separate <name>. */
foreach ($movies->movie->characters->character as $character) {
   echo $character->name, ' played by ', $character->actor, PHP_EOL;
}

?>
```

The above example will output:

- > (cmd+click);

https://www.w3schools.com/php/php_xml_simplexml_read.asp

The example below shows how to use the `simplexml_load_file()` function to read XML data from a file:

Example

```php
<?php
$xml=simplexml_load_file("note.xml") or die("Error: Cannot create object");
print_r($xml);
?>
```

Run example »

The output of the code above will be:

Display data from xml file in a table php

https://stackoverflow.com/questions/48904647/display-data-from-xml-file-in-a-table-php

0

Change your code to this:

```
<?php
$xml = simplexml_load_file("banner.xml");
foreach ($xml->student as $ban):
?>
    <tr>
      <td><?php echo $ban->Profile->Name;?></td>
      <td><?php echo $ban->Profile->ContactNumber;?></td>
      <td><?php echo $ban->Profile->Address->AddressLine1;?></td>
      <td><?php echo $ban->Profile->Email;?></td>
      <td><?php echo $ban->Profile->Picture;?></td>
    </tr>
<?php
endforeach;
?>
```

There is no `$xml->banner` in your xml file but many `student`, so I changed it to `$xml->student`.

Every `student` has a `Profile`, added this to the table.

`Address` has multiple subitems, I only added `AddressLine1` in this code, might need to modify that.

Quelle PHP:

https://www.php.net/manual/de/function.unset.php

foreach ARRAY:

https://www.php.net/manual/en/control-structures.foreach.php

Chat-GPT Prompts:

ChatGPT: Verwendet als Codeassistent um Unklarheiten in fremdem Code (Teammitglieder) zu verstehen oder um neue Möglichkeiten aufgezeigt zu bekommen wie bestimmte Probleme gelöst werden können.

Prompt Beispiele hierfür:

- Formular validieren, was muss man beachten
- foreach schleife in array die daten hizufügen
- PHP-Syntax für `echo`, das HTML-Tags mit einer PHP-Variable kombiniert.

```
echo "<table border='1'>";
```

```
echo "<tr><th>Entfernung (km)</th><th>Datum</th></tr>";

foreach ($xml->entry as $entry) {
    echo "<tr>";
    echo "<td>" . $entry->distance . "</td>";
    echo "<td>" . $entry->date . "</td>";
    echo "</tr>";
}

echo "</table>";
```

- Ich habe XML-Daten erhalten, die Fahrten repräsentieren. Jede Fahrt hat eine eindeutige ID (z. B.). wie kann ich die löchen;

```
$xml = new SimpleXMLElement($xmlString);

// Fahrt mit ID "1" löschen
foreach ($xml->fahrt as $index => $fahrt) {
    if ((string) $fahrt['id'] === '1') { // ID prüfen
        unset($xml->fahrt[$index]); // Fahrt löschen
    }
}

// Geändertes XML anzeigen
echo $xml->asXML();
```