ЛАБОРАТОРНАЯ РАБОТА №4

Обработка форм в РНР

1. Работа с глобальной переменной \$_POST

1.1. Интерпретируйте и проанализируйте следующий код

```
<div class="form">
    <form action="<?php echo $_SERVER["PHP_SELF"] ?>" method="POST">
        <fieldset>
            <legend>Оставьте отзыв!</legend>
            <div id="main_info" style="display: flex; flex-direction:</pre>
column; gap: 10px;">
                <div>
                    <label for="name">Имя:
                        <input type="text" name="name"/>
                    </label>
                </div>
                <div>
                    <label for="email">Email:
                        <input type="email" name="email"/>
                    </label>
                </div>
            </div>
            <div id="extra_info">
                <div>
                    <label for="review">Оцените наш сервис!</label>
                    <div style="display: flex; flex-direction: column;">
                        <input id="review" type="radio" name="review"</p>
value="10" checked>Хорошо
                        <input id="review" type="radio" name="review"</p>
value="8">Удовлетворительно
                        <input id="review" type="radio" name="review"</p>
value="5">Плохо
                    </div>
                </div>
            </div>
            <div id="message_info">
                    <label for="comment">Ваш комментарий: </label>
                    <textarea id="comment" name="comment" cols="30"
rows="10" class="comment"></textarea>
                </div>
            </div>
            <div id="buttons" style="display: flex; flex-direction: row;</pre>
gap: 10px; margin-top: 10px;">
                <input type="submit" value="Отправить"/>
                <input type="reset" value="Удалить"/>
        </fieldset>
    </form>
```

- 1.2. Добавьте в отмеченную область код, который будет отображать сообщение только после отправки формы.
- 1.3. Добавьте под формой функцию для проверки данных, гарантирующую заполнение всех полей и корректность введенного e-mail.
- 1.4. Объясните, что такое глобальная переменная \$_POST и \$_SERVER["PHP_SELF"]

2. Получение данных с различных контроллеров

- 2.1. Создайте форму, состоящую минимум из 3 контроллеров (input, select)
 - 2.1.1. Минимум 1 input с типом number
 - 2.1.2. Минимум 1 select
- 2.2. Тема формы определяется на ваш выбор
- 2.3. Обработайте данные и выведите их на экран

3. Создание, обработка и валидация форм

- 3.1. Создайте форму, показанную на рисунке (Рисунок 1)
- 3.2. Создайте собственную функцию **валидации**, которая будет проверять все поля формы при получении запроса
 - 3.2.1. Для поля "**name**": установите минимальную длину в 3 символа, максимальную 20 символов, и запретите использование цифр.

- 3.2.2. Для поля "**mail**": удостоверьтесь, что адрес электронной почты соответствует стандартам.
- 3.2.3. Для поля "**comment**": удостоверьтесь, что оно не пустое и укажите какиелибо другие необходимые критерии валидации (*на ваш выбор*).
- 3.2.4. Убедитесь, что пользователь отметил галочку "Do you agree with data processing?" перед отправкой формы"

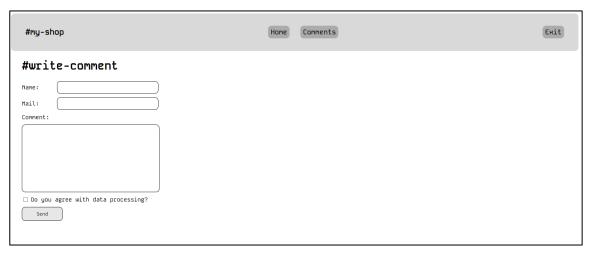


Рисунок 1. Форма комментариев

- 3.4. Если пользователь верно ввел данные, выведите комментарий ниже формы (не требуется сохранение комментариев где-либо)
- 3.5. Чем отличается глобальная переменная **\$_REQUEST** и **\$_POST**?
- 3.6. **Дополнительно** (Данное задание не является оцениваемым, а помогает вам более подробно погрузиться в изучение PHP)
 - 3.6.1. В качестве валидатора рассмотрите и изучите следующий класс.

4. Создание формы

4.1. Создайте тест из **3-х вопросов** используя input, type radio, и input, type checkbox и запросите имя пользователя. Проверьте заполнение формы и варианты, выбранные пользователем. Выведите результаты на экран.