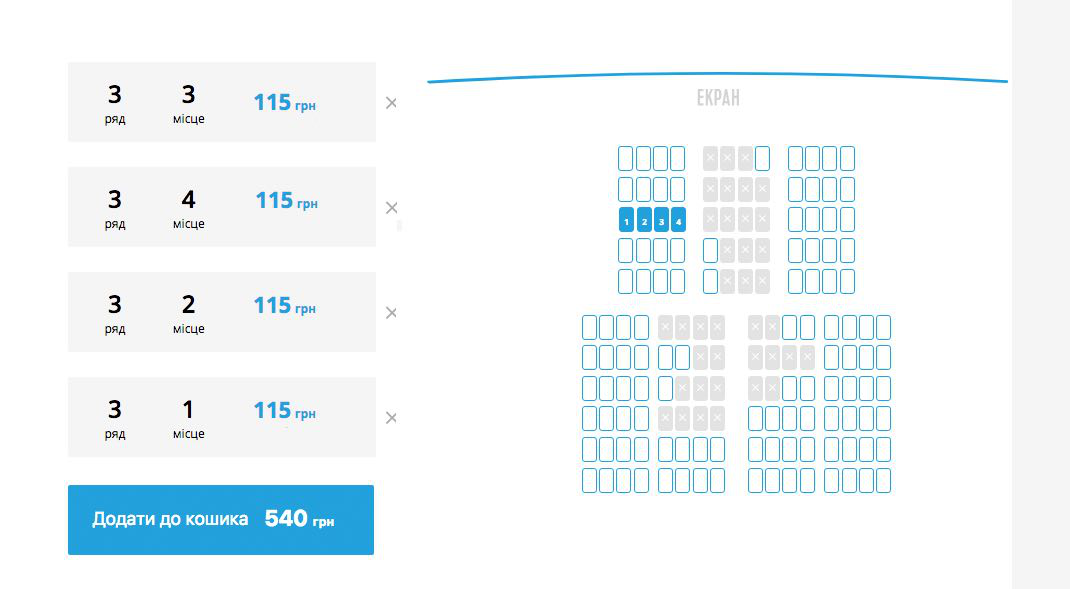
**Курсовые работы JS prof:**

Для всех работ обязательно нужно использовать модульность и паттерны!

Обязательные условия для всех курсовых.

* Модульность (Как паттерн + webpack)
* Использование ES6
* Использование рекомендуемых паттернов из предложенных к заданию

**Тема 1: Заказ билета в кинотеатре**



**Рекомендуемые паттерны:** Модуль, Обсервер, Декоратор

**Задание:** Реализовать генерацию зала с указанными параметрами, реализовать механизм покупки билетов с этого зала.

**Техническое задание:**

1. При инициализации, мы передаем объект в котором будут заданы параметры зала:

* кол-во рядов
* количество мест

**Обязательное усложнение:**

* Вместе с рядом передаем тип ряда - эконом (по умолчанию первые два ряда, обычные, премиум (последний ряд). В последствии, тип ряда будет влиять на цену билета
* Вместе с рядом передаем кол-во мест в ряду

Вызов должен выглядеть примерно так (не обязательно, реализуйте как хотите :) ):

**BuildCinema( options );**

В результате получаем сгенерированную схему зала.

2. По клику на сиденье нужно реализовать механизм “покупки” билета на это место.

Оно должно выводится в меню слева ( его номер, ряд, цена, кнопка удалить). Выбранное место в зале должно иметь подсветку активного места.

При удалении из выбранного, подсветка должна пропасть.

Под списком выбранных билетов выводится общая сумма всех билетов + таймер сколько они еще будут находится в корзине (забронированы).

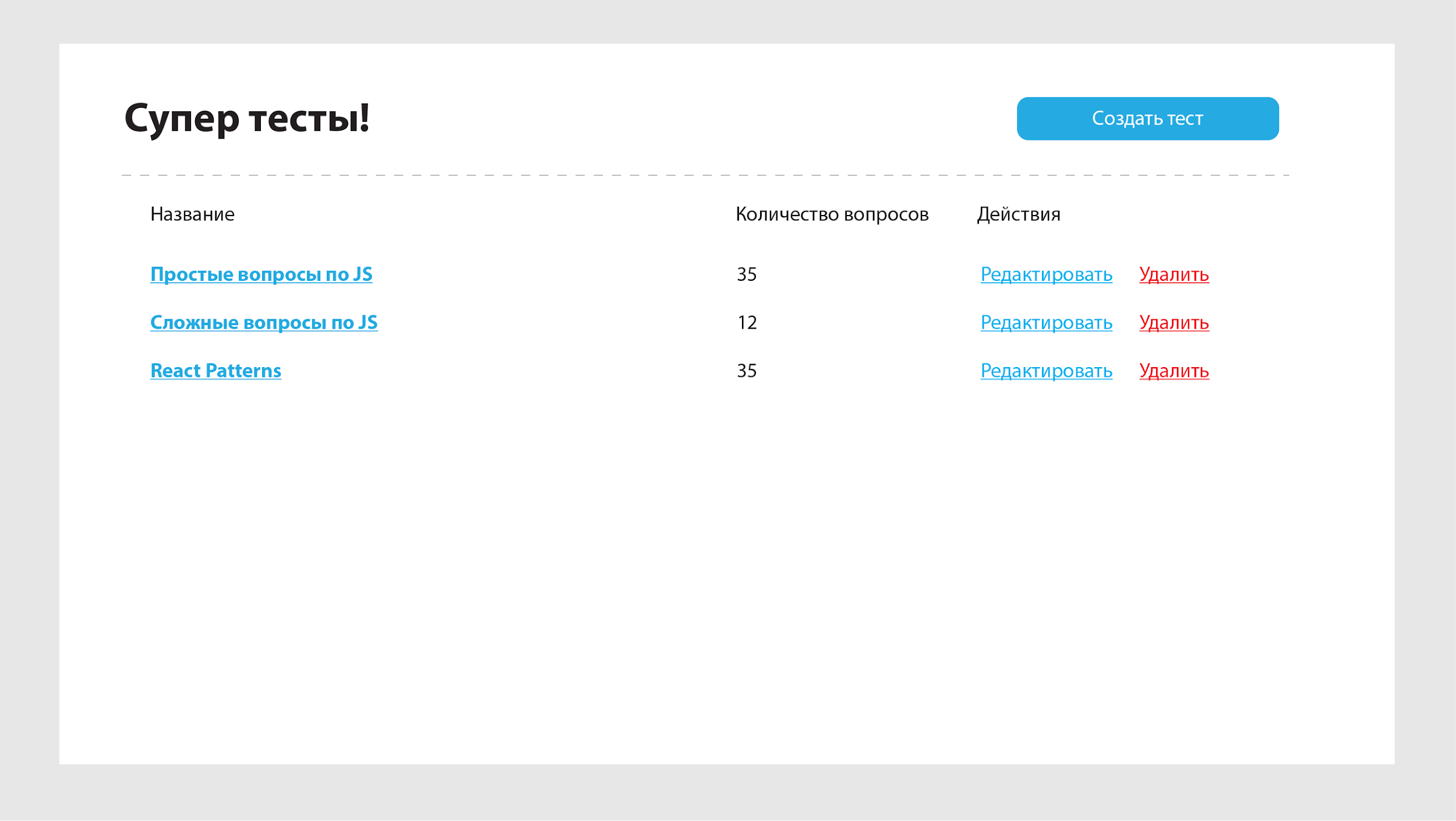
Места которые были “добавлены в корзину” находятся там 1 минуту. Если за это время добавляется еще одно место - оно обновляет счетчик. По истечению 1 минуты - билеты которые не были куплены должны стать снова не активными.

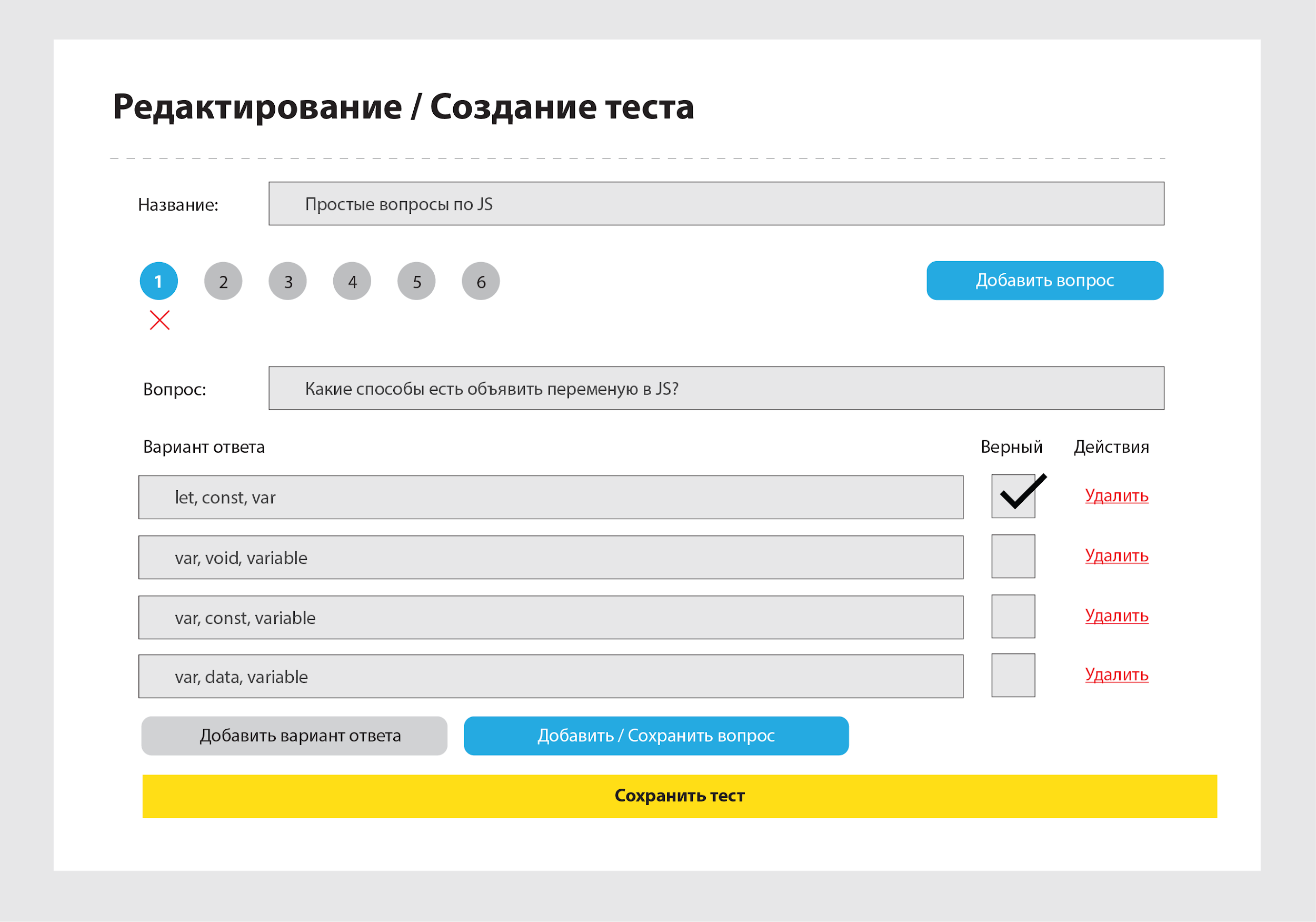
При нажатии на кнопку “Купить билеты” нужно выбранные билеты записывать в localstorage и при перезагрузке страницы показывать как проданные.

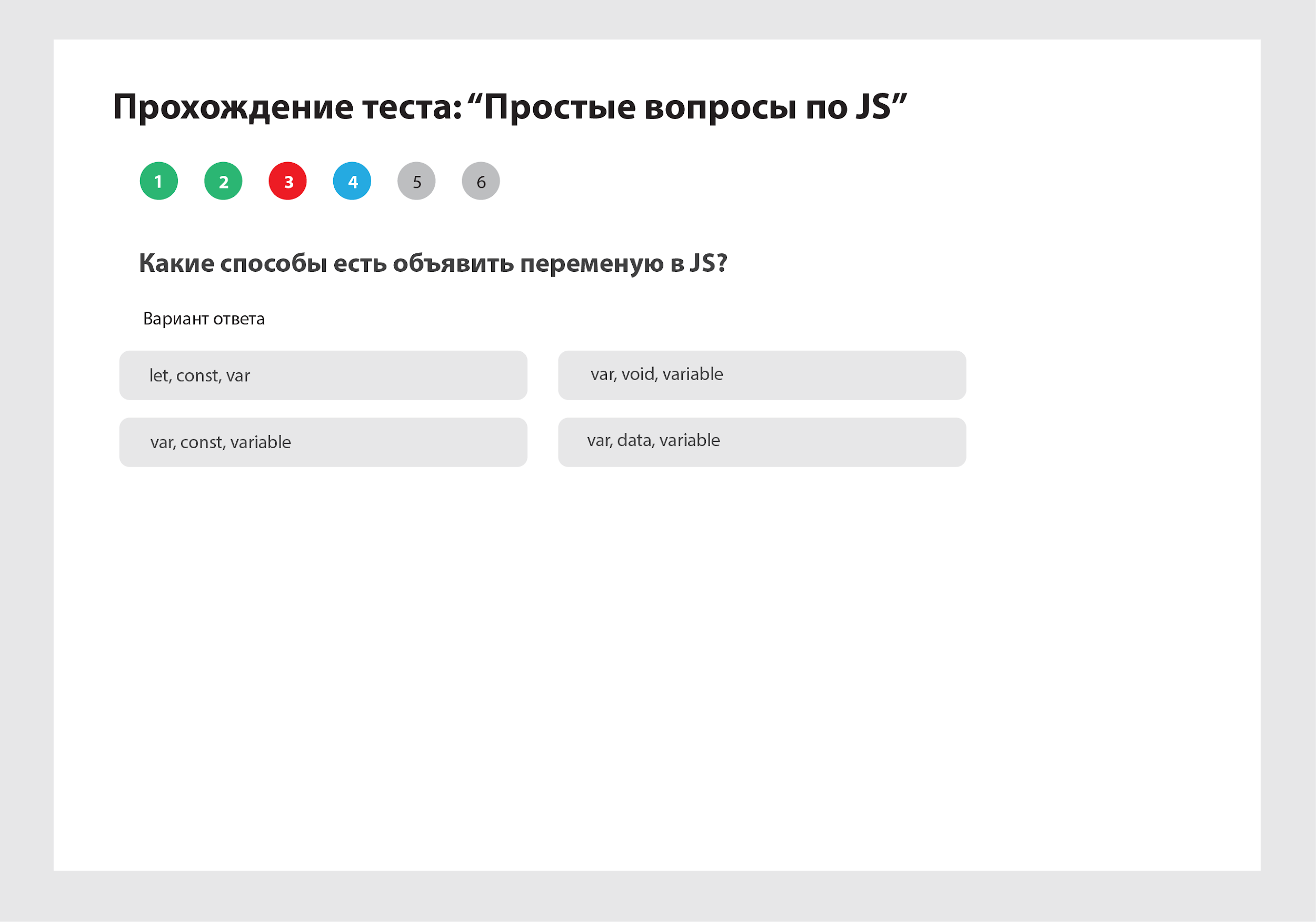
**Тема 2: Программа для тестов:**

**Рекомендуемые паттерны:** Модуль, Обсервер, Фасад

**Задание:** Написать программу, которая будет формировать список вопросов для тестов, а также механизм прохождения теста.







**Техническое задание:**

1. Создаем тест и вопросы. (Макет 2)

Вначале вводим название теста и после нажатия на кнопку “Добавить вопрос” начинаем добавлять вопросы.

В части интерфейса отвечающего за добавление вопроса, мы добавляем нужное кол-во ответов и выбираем какой из вариантов ответа будет верным.

Вариант ответа можно удалить, нажав на соответствующую кнопку.

**+ Бонус:** добавить возможность выбрать несколько вариантов ответа. По сути добавляем в программу “Тип” вопроса

Нажимая на круглые кнопки можно переключаться по вопросам. При наведении появляется крестик, при нажатии на который вопрос удаляется из теста.

Все изменения в тест и вопрос делаются по нажатию кнопки “Сохранить вопрос” и “Сохранить тест”.

**+Бонус:** Предупреждать о несохраненных изменениях.

2. Прохождение теста (Макет 1 и 3)

Выбираем тест из списка и переходим к его прохождению.

Поочередно проходим все вопросы. После выбора варианта ответа, подсвечиваем круг с номером вопроса: зеленым - если ответ верный, красным - если ошибка.

**+Бонус**. Возможность после ответа вернутся на вопрос и посмотреть какой вариант был верным или ошибочным.

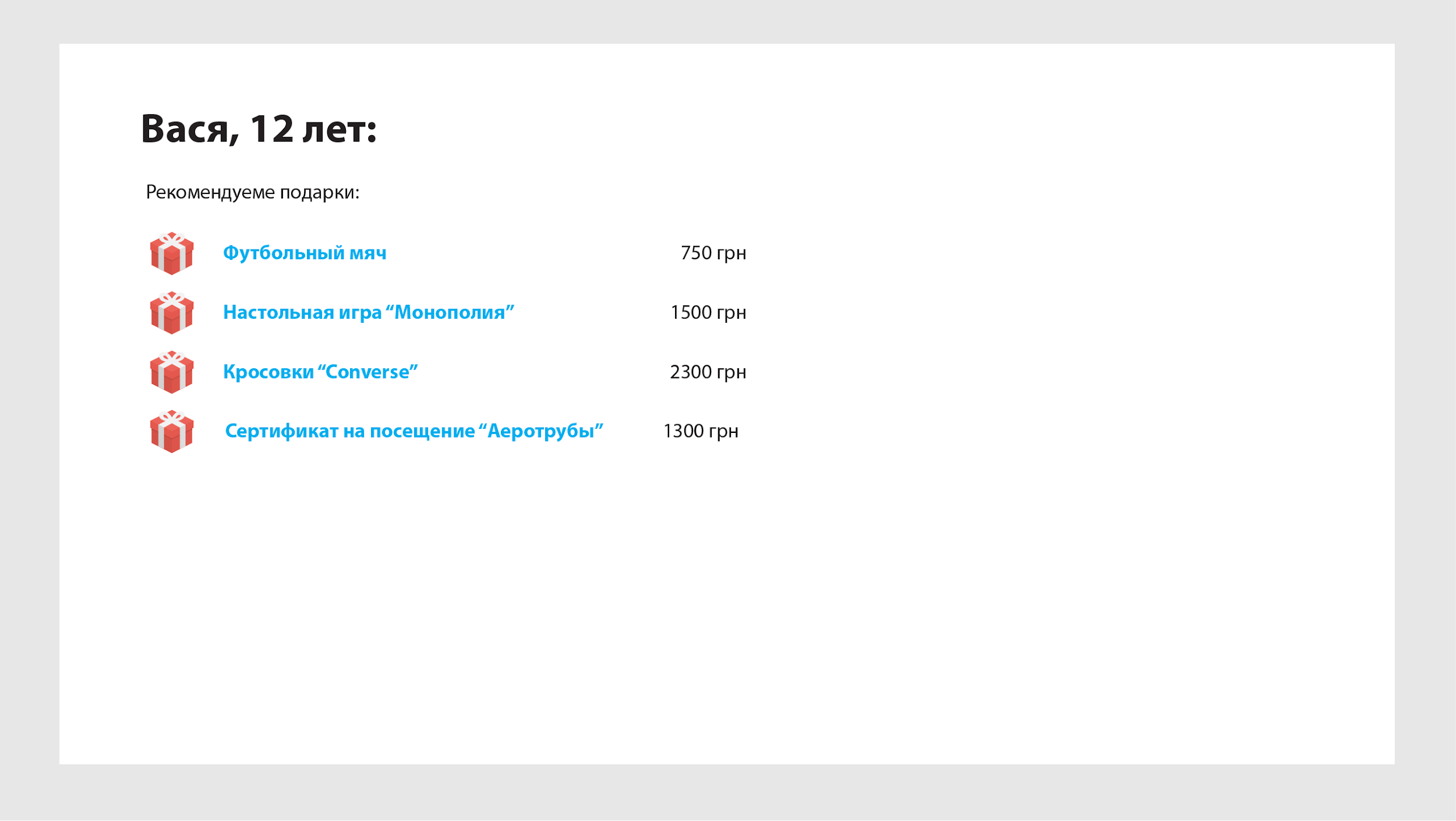
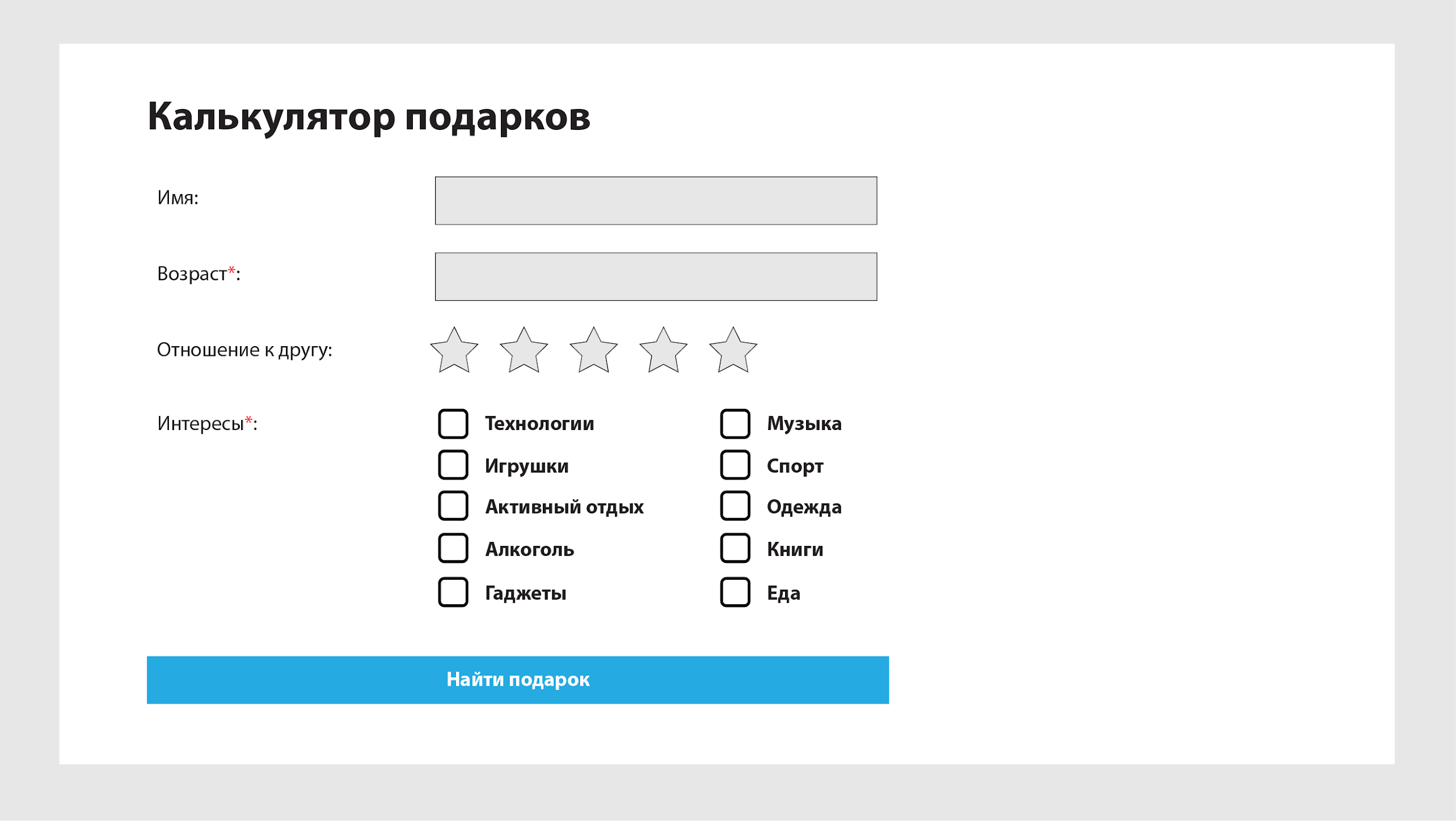
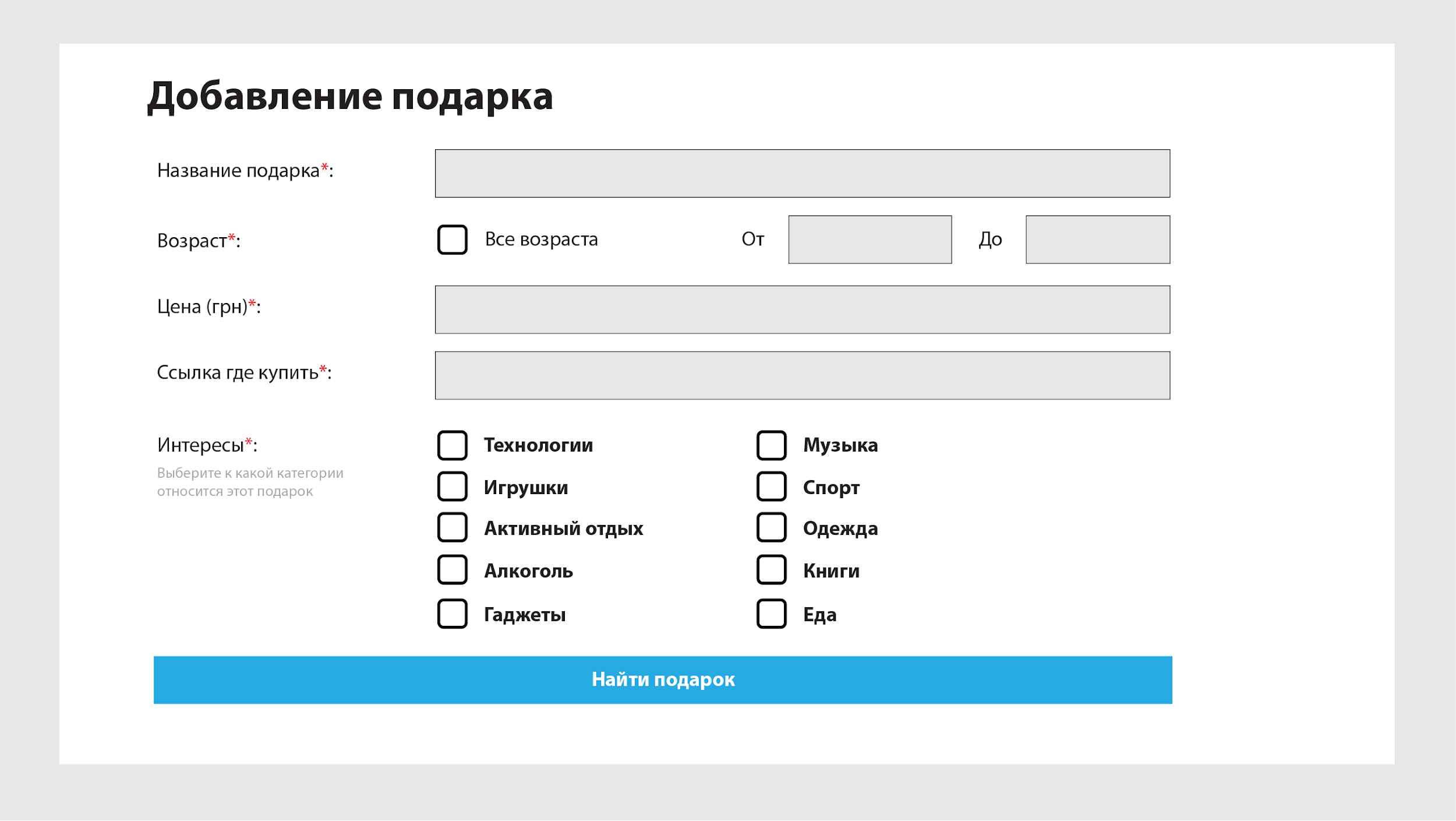
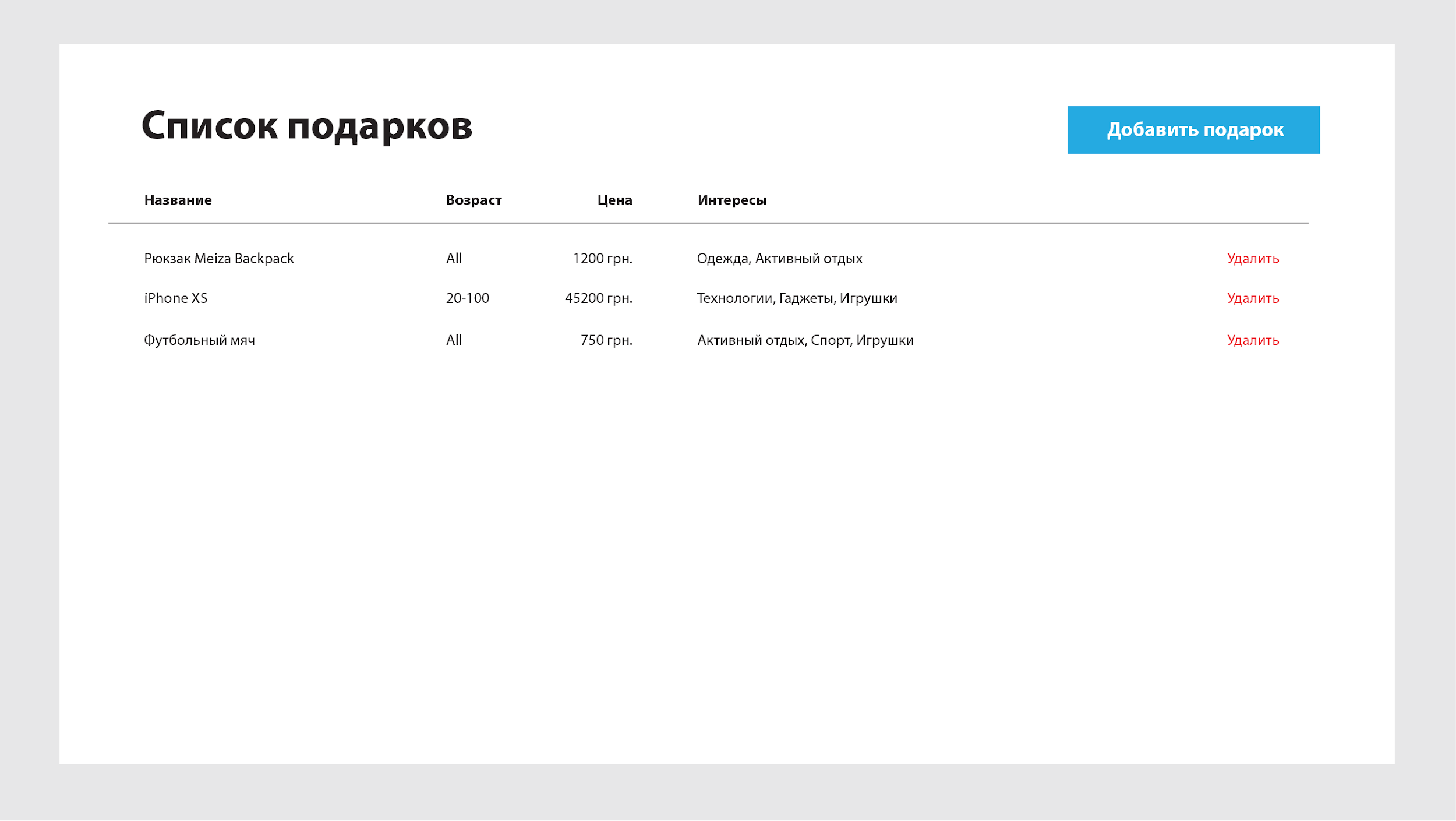
В конце выводим процент насколько хорошо вы прошли тест.

Все данные сохраняем в LocalStorage.

**Тема 3: Калькулятор подарков**

**Рекомендуемые паттерны:** Модуль, Обсервер, Команда, Декоратор

**Задание:** Написать калькулятор подарков, который предложит что подарить другу :)

****

**Техническое задание:**

1. Формируем список подарков (Макет 1)

При нажатии “Добавить подарок” показываем форму, в которой добавляем подарок. (Макет 2)

Объект который собирается из этих полей, записываем в массив подарков в localStorage.

1. Интерфейс поиска подарка.

Заполняем данные о друге, которому хотим подарить подарок. Параметр “Отношение к другу” влияет на то, насколько дорогой подарок мы будем искать.

Алгоритм поиска следующий:

* Вначале фильтруем подарки по интересам.
* После фильтруем по возрасту.
* Выдаем результаты по выбранному параметру “Отношение к другу”, если выбрано 5\5 - то показываем вверху самый дорогой подарок из найденных, если выбрано 1\5 - то показываем вверху самый дешевый.

1. Показываем окно с предложенными подарками. :)

Общее:

Показ страниц со списком и поиском можно реализовать по разному - по нажатию на кнопку, отслеживанием хеша или даже просто поднятием сервера и просто двумя разными страницами, которые работают на одном домене (localhost).

Тут реализация на ваше усмотрение.

**Тема 4: Бот для мессенджера Telegram**

**Рекомендуемые паттерны:** Любые которые вы знаете :)

**Задание:** Написать бота для телеграмма который будет работать с апи: погоды, валюты, соц.сеть или любым другим.

**Нюансы**: Для выбора этого задания вам нужны минимальные знания node.js чтобы поднять сервер и базово разобраться с [API телеграмма](https://github.com/yagop/node-telegram-bot-api).

**Тема 5: Любая ваша тема которая хоть как-то по сложности напоминает эти задания.**