

# Задача

Реализовать одностраничное (SPA) демо приложение с использованием React по покупке авиабилетов.

## Описание логики приложения

В приложении присутствуют три главных состояния, которые показаны на скриншотах ниже:

1. **Начальное состояние приложения.** Отображается интерфейс поиска билетов. Пользователь должен ввести пункт отправления, пункт назначения и установить максимальный вес багажа. Далее приложение должно начать поиск по заданным параметрам и выдать список подходящих билетов.
1. **Поисковая выдача.** Отображается список билетов удовлетворяющий критериям поиска и пустая область - корзина (Полный список билетов можно найти в файле JSON в прикрепленном архиве). Добавление билета в корзину осуществляется кликом по билету, удаление - повторным кликом. После добавления билета в корзину она должна отобразить свое новое состояние.
1. **Состояние корзины и покупка.** В корзине отображаются выбранные при первом поиске билеты, но мы можем изменить критерии поиска, найти новые билеты и также добавить их в корзину. Корзина при этом должна содержать в себе все выбранные ранее билеты. По нажатию на кнопку "Купить" данные корзины должны быть сгруппированы в JSON и быть готовы к отправке на сервер (в качестве данных билета можно указывать его ID плюс желаемый объем багажа, однако в данном случае цены фиксированные и багаж не будет влиять на стоимость, но и приложение тестовое). После формирования JSON можно просто вывести его в консоль.

Скриншоты итогового приложения:

Откуда



Санкт-Петербург

Куда



Москва

Багаж (кг)



20



Начните поиск билетов

Откуда



Санкт-Петербург

Куда



Москва

Багаж (кг)



20



08:20

Санкт-Петербург  
31.12.18

7ч

40 900 р

12:20

Москва  
31.12.18

08:20

Санкт-Петербург  
31.12.18

7ч

40 900 р

12:20

Москва  
31.12.18

08:20

Санкт-Петербург  
31.12.18

7ч

40 900 р

12:20

Москва  
31.12.18

08:20

Санкт-Петербург

7ч

40 900 р

12:20

Москва



Корзина пуста

Откуда

Санкт-Петербург

Куда

Москва

Багаж (кг)

+

20

−

08:20

Санкт-Петербург  
31.12.18

7ч

≈

12:20

Москва  
31.12.18

40 900 р

08:20

Санкт-Петербург  
31.12.18

7ч

≈

12:20

Москва  
31.12.18

40 900 р

08:20

Санкт-Петербург  
31.12.18

7ч

≈

12:20

Москва  
31.12.18

40 900 р

08:20

Санкт-Петербург  
31.12.18

7ч

≈

12:20

Москва  
31.12.18

40 900 р

Санкт-Петербург

Москва

2 билета на сумму 81 800

Париж

Лондон

2 билета на сумму 120 000

Вашингтон

Оттава

2 билета на сумму 81 800

Купить

## Требования

1. Приложение должно быть написано на React, собираться в Webpack.
2. Приложение должно содержать единый store для данных (продумать какие данные должны в этот store попадать, а какие могут храниться локально в конкретном компоненте). В качестве store может выступать либо state родительского компонента приложения, либо Redux по желанию.
3. Имитация запросов к API может быть реализована через `setTimeout` обернутый в `Promise` чтобы имитировать загрузку, а также обработку положительных результатов и возможных ошибок.
4. Требований по оформлению кода нет. Однако код должен быть оформлен аккуратно и полностью соответствовать вашим файлам настроек линтеров. Документирование и комментирование кода приветствуется.

5. Требований по написанию стилей тоже нет. Однако они также должны соответствовать stylelint при его наличии. Можно использовать как чистый CSS так и любой препроцессор LESS/SCSS/SASS. Явное именование классов, CSS Modules, Styled Components не имеет значения, используйте то, что удобно.
6. Правильная семантика тегов внутри компонентов.
7. Т.к. макеты даны в формате png (а не sketch или PSD), то строгое соблюдение дизайна не требуется, но стараться максимально его придерживаться.

По логике работы приложения:

1. Корректное отображение различных статусов приложения (начальное, отсутствие билетов, загрузка билетов, состояние корзины и тд.)
2. Обработка потенциальных ошибок.

## Пожелания:

1. Использование SSR
2. Использование CSS Modules

## JSON с тестовыми данными:

```
[
  {
    "id": 1,
    "from": "Санкт-Петербург",
    "to": "Москва",
    "time_departure": "20-05-2018 8:20",
    "time_arrival": "20-05-2018 9:20",
    "price": 40000,
```

```
    "luggage_max": 5,
    "duration": 3
  },
  {
    "id": 2,
    "from": "Санкт-Петербург",
    "to": "Москва",
    "time_departure": "21-05-2018 9:20",
    "time_arrival": "21-05-2018 10:20",
    "price": 50000,
    "luggage_max": 8,
    "duration": 3
  },
  {
    "id": 3,
    "from": "Санкт-Петербург",
    "to": "Москва",
    "time_departure": "22-05-2018 9:20",
    "time_arrival": "22-05-2018 10:20",
    "price": 60000,
    "luggage_max": 12,
    "duration": 3
  },
  {
    "id": 4,
    "from": "Париж",
    "to": "Лондон",
    "time_departure": "22-05-2018 9:20",
    "time_arrival": "22-05-2018 17:20",
    "price": 60000,
    "luggage_max": 12,
    "duration": 8
  },
  {
    "id": 5,
    "from": "Париж",
    "to": "Лондон",
    "time_departure": "22-05-2018 9:20",
    "time_arrival": "22-05-2018 17:20",
    "price": 20000,
    "luggage_max": 4,
    "duration": 8
  },
  {
    "id": 6,
    "from": "Париж",
```

```

    "to": "Лондон",
    "time_departure": "10-05-2018 9:20",
    "time_arrival": "10-05-2018 17:20",
    "price": 65000,
    "luggage_max": 6,
    "duration": 8
  },
  {
    "id": 7,
    "from": "Вашингтон",
    "to": "Оттава",
    "time_departure": "22-05-2018 9:20",
    "time_arrival": "22-05-2018 17:20",
    "price": 60000,
    "luggage_max": 5,
    "duration": 15
  },
  {
    "id": 8,
    "from": "Вашингтон",
    "to": "Оттава",
    "time_departure": "10-05-2018 9:20",
    "time_arrival": "10-05-2018 17:20",
    "price": 50000,
    "luggage_max": 5,
    "duration": 16
  },
  {
    "id": 9,
    "from": "Вашингтон",
    "to": "Оттава",
    "time_departure": "22-05-2018 9:20",
    "time_arrival": "22-05-2018 17:20",
    "price": 90000,
    "luggage_max": 5,
    "duration": 16
  }
}

```