# **Часть 1**

В первой части проекта тебе предстоит протестировать функциональность каршеринга.

**1. Подготовь тестовую документацию, чтобы проверить вёрстку формы бронирования**

1. Изучи макеты каршеринга и определи объекты тестирования: это все визуальные элементы на макете, которые нужно проверить.
2. Составь чек-лист, по которому будешь тестировать вёрстку формы бронирования.
   * Выбери только один тариф. Например, «Роскошный».
   * Проверь орфографию, расположение элементов, их внешний вид — в том числе на навигационной карте, которая находится в центральной части интерфейса. Валидацию полей тестировать не нужно.
   * Проверяй только саму форму бронирования. Вёрстку окон «Добавление прав», «Способ оплаты», «Добавление карты» тестировать не нужно.

Помести чек-лист на первую вкладку таблицы — «Часть 1. Чек-лист на вёрстку».

**2. Подготовь тестовую документацию, чтобы проверить логику работы**

Проанализируй требования к функциональности «Каршеринг» и подготовь:

1. Чек-лист, по которому будешь проверять функциональность окон «Способ оплаты» и «Добавление карты».
2. Тест-кейсы на логику работы кнопки «Забронировать» и логику функциональности бронирования — см. пункт требований «Бронь машины».

Обрати внимание:

* На кнопке «Забронировать» указаны расстояние и время в пути. Например: «Маршрут составит 3 км и займёт 4 мин». Тебе не нужно проверять, правильно ли рассчитаны эти данные.
* Таймер, который отсчитывает время бесплатного ожидания, тестировать также не нужно.

Когда будешь составлять чек-лист, используй технику разбиения на классы эквивалентности и выделения граничных значений.

Не забудь про позитивные и негативные проверки.

Напиши чек-лист во второй вкладке таблицы — «Часть 1. Чек-лист «Способ оплаты» и «Добавление карты».

Оформи тест-кейсы на третьей и четвёртой вкладках таблицы:

* «Часть 1. Тест-кейсы на кнопку «Забронировать».
* «Часть 1. Тест-кейсы на логику функциональности бронирования.

**3. Протестируй приложение и заведи баг-репорты**

Времени на тестирование осталось мало, поэтому проверь новую функциональность Яндекс.Маршрутов только в двух конфигурациях окружения:

* в Яндекс.Браузере при разрешении экрана 800x600;
* в Firefox при разрешении экрана 1920x1080.

Протестируй функциональность каршеринга по чек-листам и тест-кейсам, которые тебе удалось спроектировать ранее.

Тестирование вёрстки нужно провести в обоих окружениях. Логику приложения достаточно проверить в одном окружении.

Обрати внимание: в требованиях сказано, что после заполнения поля «Добавить права» включается таймер на 30 секунд. В текущей реализации его нет. Разработчики специально передали тестовый стенд: так тебе не придётся ждать каждый раз, когда выполняешь проверку.

Когда будешь тестировать, отмечай результаты выполнения проверок: PASSED или FAILED. Если тест со статусом FAILED — заведи баг-репорт в Яндекс.Трекере в [очереди BUG](https://tracker.yandex.ru/bug). Не забудь вписать ID в соответствующую таблицу результатов.

В названии документа напиши имя, фамилию, номер когорты и номер спринта, для которого выполняешь задание. Например: «Иван Иванов, 2-я когорта — 2-й спринт».

4**. Подведение итогов (\*)**

В будущем тебе придётся часто давать обратную связь по результатам работы. Чтобы потренироваться, можешь выполнить дополнительное задание. Ревьюер проверит его и даст обратную связь. Это не повлияет на зачёт.

* Поделись, какое впечатление оставили у тебя Яндекс.Маршруты как у пользователя?
* Расскажи в нескольких предложениях, что именно тебе удалось протестировать?
* Подведи итоги тестирования: например, тебе удалось провести все тесты и найти несколько багов. Приложи ссылки на них, чтобы коллеги могли их быстро открыть и посмотреть.
* Сделай вывод: как ты думаешь, можно ли отдать такой продукт пользователям? Помни, что тестировщик даёт рекомендации, а итоговое решение принимает команда менеджеров.

### **Как будут оценивать работу**

Твою работу будут проверять по следующим критериям:

* хорошо ли оформлены тест-кейсы и какова полнота проверок;
* хорошо ли оформлены чек-листы и насколько полно спроектированы проверки;
* качественно ли проведено тестирование: выполнены ли все проверки, которые удалось спроектировать;
* какие баги заведены и насколько корректно они оформлены.

https://code.s3.yandex.net/qa/schemes/project-roz.png

# **Часть 2**

Тебе предстоит протестировать новый вид транспорта в Яндекс.Маршрутах — аэротакси.

### **Постановка задачи**

Частные авиаперевозки развиваются быстро, и нужно успеть занять нишу. Поэтому в Яндекс.Маршруты добавили новый вид транспорта — аэротакси.

Чтобы ускорить разработку, фронтенд и бэкенд для этой функциональности делали одновременно: фронтенд уже готов, а бэкенд задерживается.

Тебе предстоит протестировать реализацию на фронтенде, не дожидаясь бэкенда.

Пока что достаточно проверить функциональность только в Яндекс.Браузере.

### **Технические данные**

В процессе разработки договорились, что структура ответа для аэротакси будет такой же, как и для остальных видов транспорта.

Значения такие:

{

"id": "aero",

"name": "Аэротакси",

"icons": {

"inactive": "helicopter.svg",

"active": "helicopter-active.svg"

}

}

Логика расчётов для аэротакси такая же, как и для остальных видов транспорта.

Рассчитанная стоимость должна возвращаться с бэкенда в таком формате:

"aero": {

"price": 3000,

"duration": 0.1

}

### **Как выполнить задание**

Создай чек-лист со списком проверок для нового вида транспорта. Помести результат в последнюю вкладку шаблона: «Часть 2. Чек-лист на Аэротакси».

В чек-листе будут две проверки, с которыми могут возникнуть трудности.

1. **Проверить, что аэротакси появилось в интерфейсе приложения**

Открой Charles и перехвати ответ бэкенда, который возвращает виды транспорта. Вставь в ответ новый блок с аэротакси:

{

"id": "aero",

"name": "Аэротакси",

"icons": {

"inactive": "helicopter.svg",

"active": "helicopter-active.svg"

}

}

Чтобы понять, куда его вставить, — изучи, как расположены блоки для других видов транспорта в теле ответа бэкенда.

1. **Проверить, что расчёт стоимости для аэротакси отобразился в интерфейсе**

Теперь на фронтенде должен отображаться новый вид транспорта. Перехвати ответ бэкенда, в котором содержатся расчёты стоимости, и подставь в него блок с аэротакси:

"aero": {

"price": 3000,

"duration": 0.1

}

Чтобы понять, куда его вставить, — изучи, как расположены блоки с расчётами для других видов транспорта в теле ответа.

На базе изменённых ответов создай автоматический ответ — так удобнее тестировать приложение.

Когда настроишь все автоматические ответы, протестируй новую функциональность по подготовленному чек-листу. Если нужно — заведи баг-репорты.

Вспомни про распространённые [ошибки](https://code.s3.yandex.net/qa/files/charles-faq.pdf) в Charles.

### **Оформление**

Создай папку на Гугл-диске или Яндекс.Диске. Помести туда:

1. Скриншот из DevTools или Charles с оригинальным запросом видов транспорта.
2. Скриншот с изменённым запросом видов транспорта из Charles.
3. Скриншот окна настройки автозамены запроса видов транспорта в Charles.
4. Скриншот из DevTools или Charles с оригинальным запросом расчёта стоимости и длительности поездки.
5. Скриншот с изменённым запросом расчёта стоимости и длительности поездки из Charles.
6. Скриншот окна настройки автозамены с запросом расчёта стоимости и длительности поездки в Charles.

Заполни чек-лист со ссылками на баг-репорты, если они будут. Помести результат в последнюю вкладку шаблона: «Часть 2. Чек-лист на Аэротакси».

### **Как будут оценивать работу**

Задания будут оценивать по следующим критериям:

* хорошо ли оформлены чек-листы и насколько полно спроектированы проверки;
* качественно ли проведено тестирование: выполнены ли все проверки, которые удалось спроектировать.

https://code.s3.yandex.net/qa/schemes/project-roz.png

## **Как сдать проект**

Помести результаты обеих частей проекта в гугл-док. Для этого создай копию [шаблона](https://docs.google.com/document/d/1PMXeSXJJYUSbMpN7NGCab_NhOkdJkYXeoD5G3skkDt4/edit) и заполни её. Открой доступ на комментирование по ссылке и прикрепи её в тренажёр — во вкладку «Ревью». Вкладка появится в следующем уроке.

## **Дождись проверки и комментариев**

Ревьюер проверит работу в течение 48 часов. Если будут замечания, проект вернётся тебе на доработку. Доработать задания и прислать их на проверку ещё раз можно в течение 48 часов.