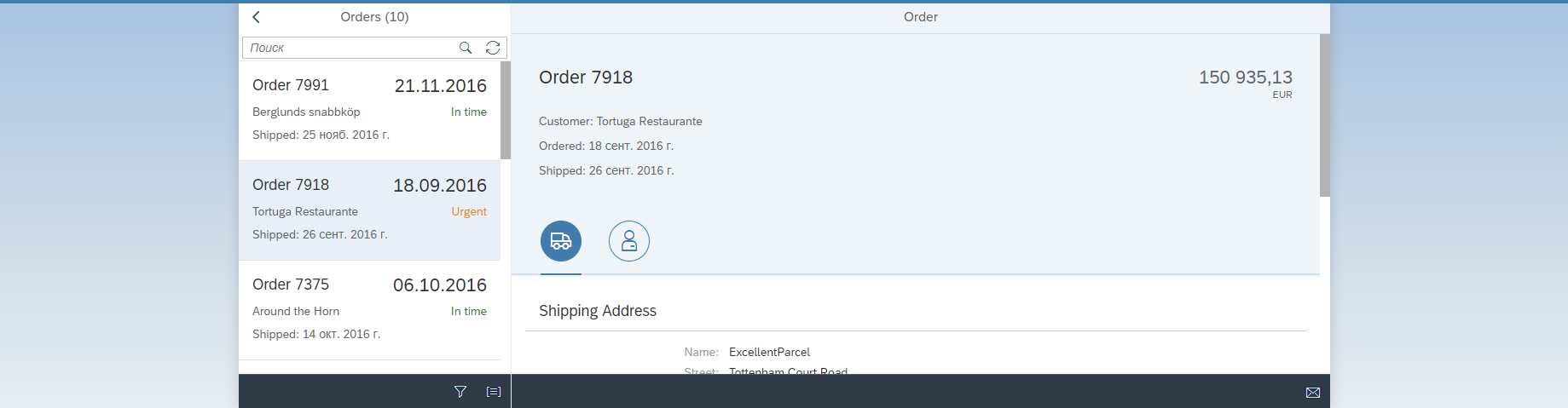
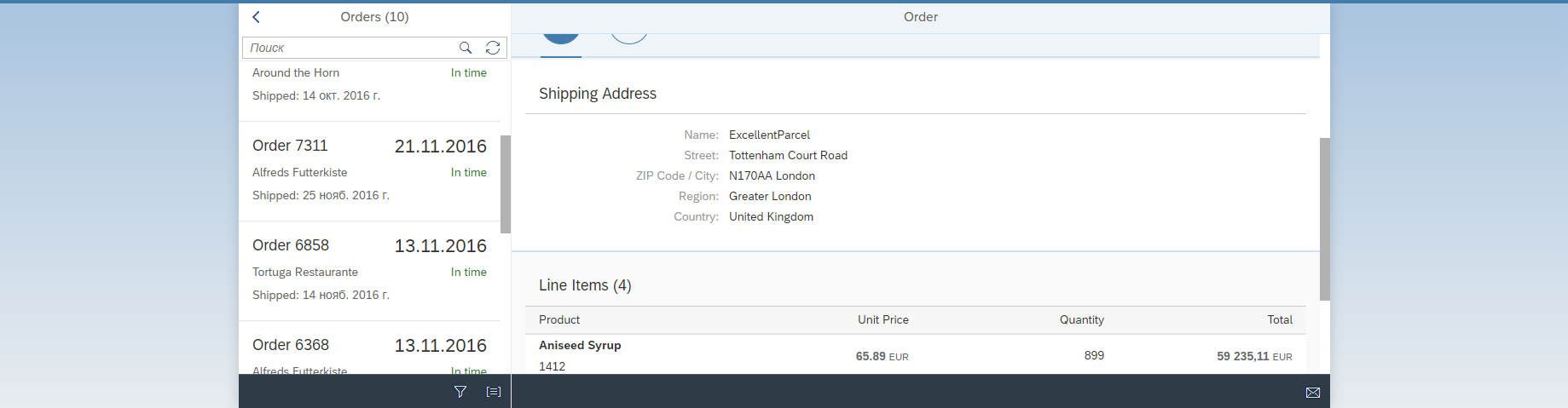
## 1 - Визуальная часть ([Demo video страницы](https://drive.google.com/file/d/12jewUDwnNMLF0XwMaN7GPD3pBPgeTTdW/view?usp=sharing))

1. Проверить корректное отображение и доступность футеров при уменьшении высоты экрана:

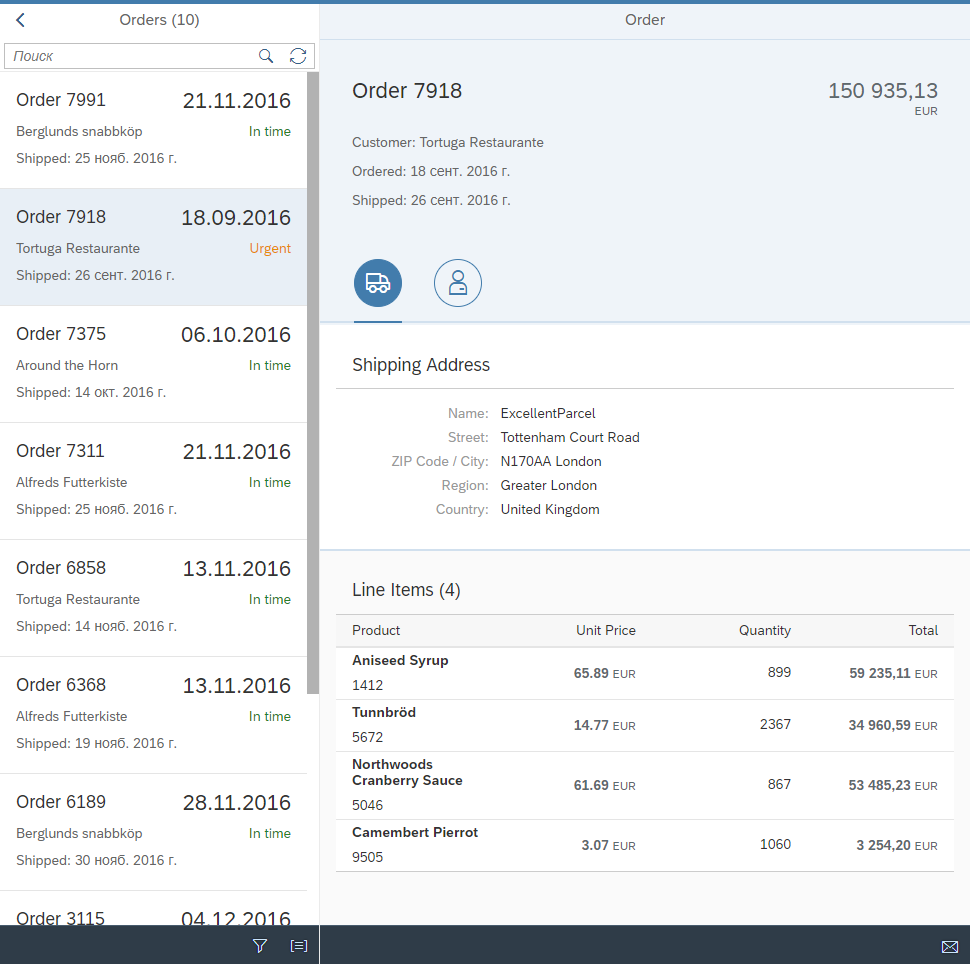




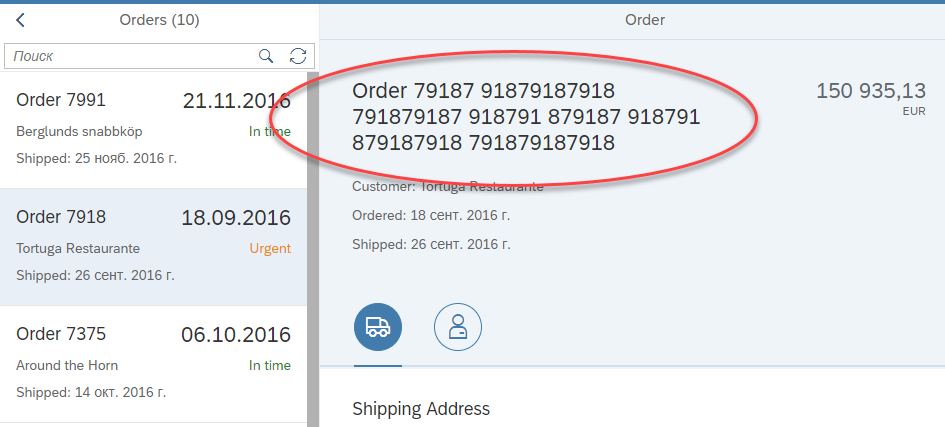
Обе части страницы (блок заказов и блок деталей заказа) должны scroll’иться независимо друг от друга. Скроллбары должны появляться лишь тогда, когда список заказов или детальная информация не помещаются в доступное вертикальное пространство.

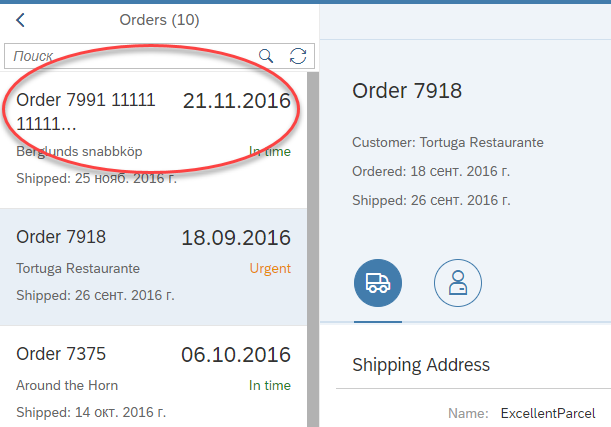
Также, следует обратить внимание на заголовок правой части (детали заказа), при скроллинге деталей, заголовок должен оставаться прибитым кверху, это более user-friendly.

2. Проверить корректное отображение страницы при сужении ширины экрана:

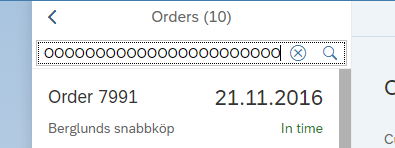


3. Проверить корректное отображение “длинных текстов”. В редких случаях текстовые поля могут занимать больше места чем этого ожидаешь, но приложение должно быть к этому готово:





4. При переполнении поля поиска текстом, убедиться что текст не наслаивается на иконки справа:



5. При наведении на кликабельные части приложения, тип курсора нужно сменить на “pointer”. Исключение - поля для ввода, там должен оставаться стандартный курсор типа “text”;

6. При наведении курсора на элемент заказа - его фон нужно подсветить, так будет проще пользователю “найти свой курсор” и идентифицировать элемент, который после клика будет выбран.

## 

## 2 - Программная часть

1. Названия идентификаторов (class, id, variable, function etc.) должны быть:

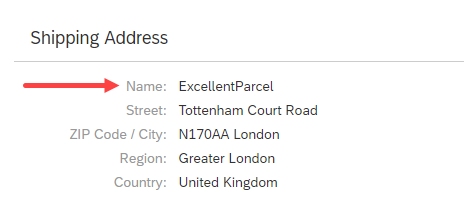
1. Лаконичными;
2. Отражать суть контента (описывать предметную область);
3. Не должны быть слишком общими (“table”, “form”, “data” и т д.);
4. На английском языке (и не использовать транслитерацию);
5. **Classes:** В стиле hyphen-separated (разделяем отдельные слова дефисами).
6. **Classes:** Имя существительное (e.g. ~~“wrap-box”~~ “box-wrapper”) или связка прилагательное-существительное (e.g. “selected-order”)

2. Использовать селекторы по тегу - плохо. Так как стили, прописанные в них, будут влиять абсолютно на все теги на странице (беда в том, что для нужд конкретной страницы ты переписываешь дефолтные значения тегов).

* css из этой страницы, возможно, понадобиться переиспользовать на другой. И тогда элементы из новой страницы получат нежелательные стили;
* эта страница может расшириться (добавятся новые секции, элементы интерфейса) и возможно, новым элементам не понадобятся эти общие стили, и тогда, где они не нужны, их придётся сбрасывать.

3. Не нужно заносить application-related логику в CSS. Обычно приложения - интернациональные. Есть специальный конфигурационный файл, который хранит переводы полей под каждый язык. При таком подходе, для того чтоб перевести приложение на русский язык, придётся менять CSS - что очень плохо. CSS - это технология для стилизации.

Как пример: занесение лейблов элементов формы в псевдоэлемент ::before - не самая лучшая затея.



4. Желательно, чтоб заголовок страницы (title), отражал её суть.

5. Не заполнять страницу “искусственными данными”. Желательно сделать как на мокапе, так будет максимально приближено к реальности и лучше смотреться. В крайнем случае, использовать “рыбу” по типу *lorem ipsum.*.

6. Убрать всё, что не используется (не привносит никакого функционала или стилистики):

* Атрибуты в HTML;
* CSS правила;
* you name it..

7. Желательно не использовать стилизацию атрибутами. Для этого есть CSS.

8. Inline styles - это плохо. Для этого есть CSS. Используя inline-styles мы смешиваем CSS (presentation layer) и HTML (data/semantics layer), это усложняет читабельность кода, дебаг и его поддержку в целом. К inline-style так же можно отнести установку внешнего вида через HTML атрибуты, в идеале и этого избегать.

9. Было бы хорошо разбить большой css файл на логические области и разделить их поясняющими комментариями. Или же разбить на отдельные файлы.

10. В комментариях к коду использовать only english language. Представь свои чувства, когда тебе достанется код, который писал китаец и использовал родной язык для комментариев.

11. Не выравнивать текст через “&nbsp;”, скорее всего эту задачу можно решить используя CSS.

12. Не нужно включать в коммит лишние файлы, те что создаёт IDE для проекта.

13. Использовать inline элементы по их назначению - отображение/форматирование текста.

14. По возможности использовать shorthand notation для установки значений CSS свойств.

15. Всегда помнить о том, что страница будет открываться пользователями на разных устройствах, т.е. размеры экрана будут отличаться от размеров твоего монитора.

16. Не нужно оставлять закомментированный код без веской причины. Перед сдачей кода в prod или на проверку, в этом нет смысла, только сбивает с толку людей, который будут потом поддерживать этот код.

17. Не делать, чего не просят. Пока мало опыта, не пытаться предугадать желание заказчика, скорее всего они не сойдутся :) Лучше просто уточнить/переспросить.

Когда появится опыт и будет очевидно, что требование заказчика или части мокапа выглядят и ведут себя странно или не user-friendly, всегда высказывать своё мнение/предложения по улучшению. В большинстве случаев заказчик не знает чего хочет, наша задача - дать ему это понять.

18. Использовать TAB для индентации, выставить размер TAB’а в 4 space’a (выставляется в настройках IDE).

19. Если назначается класс, который будет использоваться только для работы с элементом через js, то префексируй его словом “js”: “js-order-item”. Ключевое слово “**только**”, если мы его завели для скрипта, то стилизацию он больше не несёт.

20. Определиться, какой стиль кавычек использовать одинарные ‘ или двойные “ и использовать их везде (желательно двойные). Код должен быть консистентен.

21. Не использовать <table> в целях выравнивания. Таблицы создавались для отображения “табличных” данных, где есть несколько колонок и количество строк либо большое, либо динамически меняется.

22. Не начинать комбинированный селектор с названия тега: **div.my-order**. Это неэффективно и скорее всего, бессмысленно. Почитать [тут](https://css-tricks.com/efficiently-rendering-css/" \l "article-header-id-2). Желательно прочитать всю статью.

23. SVG выносить в файлы (не делать вставки текстовых массивов в HTML, это захламляет его).

24. Стараться использовать семантические теги.