**Figma**

Figma — это графический онлайн-редактор для дизайнеров интерфейсов и веб-разработчиков. Сейчас это удобная бесплатная альтернатива Photoshop. Большое преимущество платформы — возможность работать прямо в браузере. При этом есть и десктопная версия. Расскажем, что надо знать верстальщику при работе с макетом в Figma.

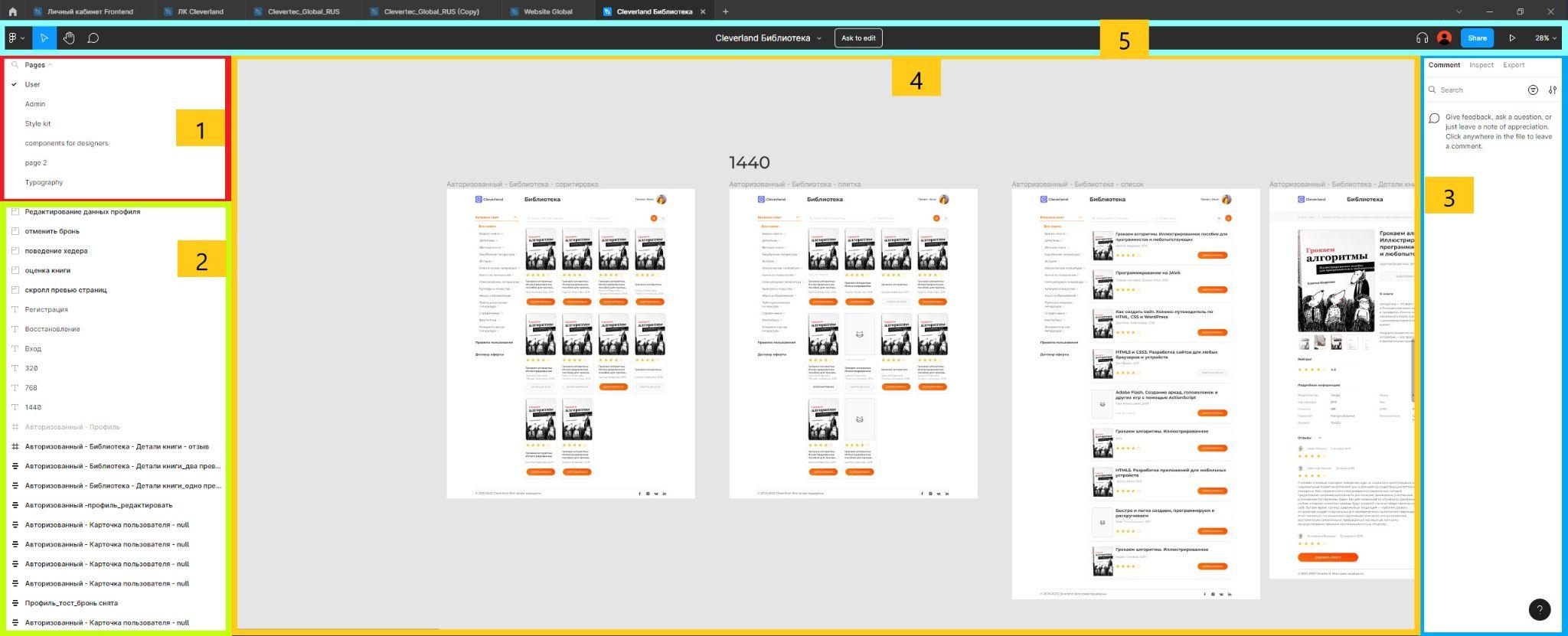
[Ссылка на макет](https://www.figma.com/file/d8LhhLjMkaTfPvAcYQULNv/Library---students-file?node-id=3601%3A32182&t=jUXbr7cMFW80wO7J-0)

## 

## **Структура макета и рабочие инструменты**

Интерфейс в разделе работы с макетом состоит из нескольких блоков.

1. В боковой панели слева показана вся структура проекта. Можно осуществить навигацию по листам (поз. 1) Также можно проследить вложенность элементов и найти нужный (поз. 2). Каждый тип элемента обозначен отдельной иконкой — текст, изображение, составной блок.
2. Основная рабочая область в центре (поз. 4) — для непосредственного взаимодействия с макетом.
3. В верхней части страницы расположено основное меню (поз. 5). Большая часть инструментов в нём предназначена для вёрстки.
4. Боковая панель справа (поз. 3) включает в себя три вкладки. В этих вкладках находится вся доступная информация об элементе.



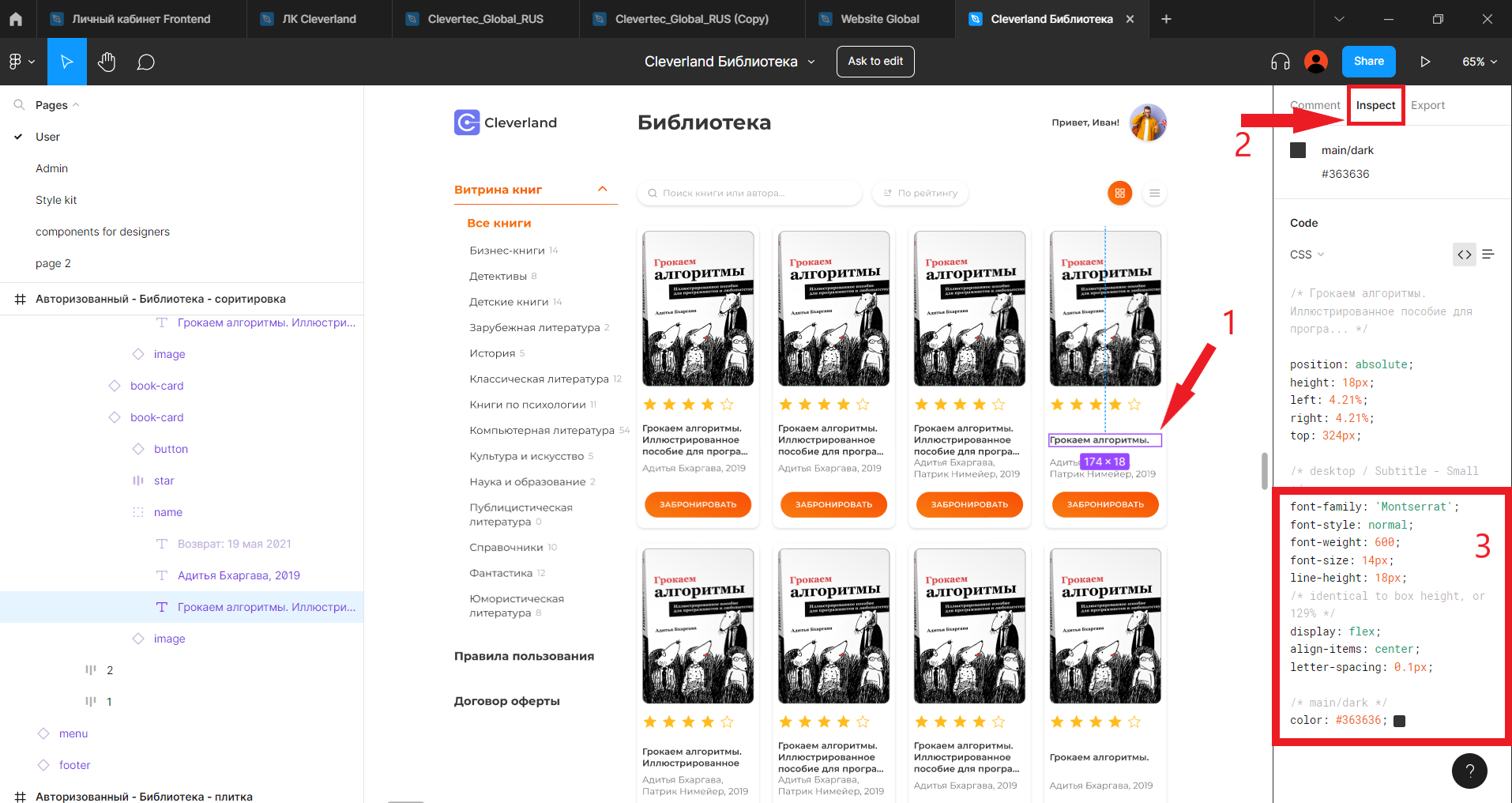
Интерфейс страницы работы с проектом

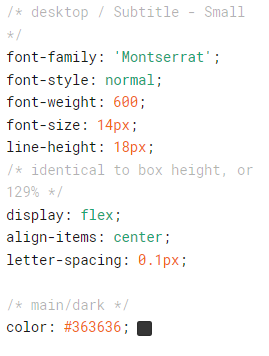
## **Работа с параметрами элементов**

Основная задача верстальщика при работе с макетом — получить параметры элементов. Давайте разберём по пунктам, как это сделать в Figma.

### **Текст**

Чтобы узнать все текстовые параметры, нужно выделить элемент (шаг 1) и открыть вкладку Inspect) в правой боковой панели (шаг 2. Там в списке свойств отобразятся все используемые параметры — название, размер, насыщенность и цвет шрифта, высота строки и остальные (шаг 3).





Работа с текстовыми элементами

### 

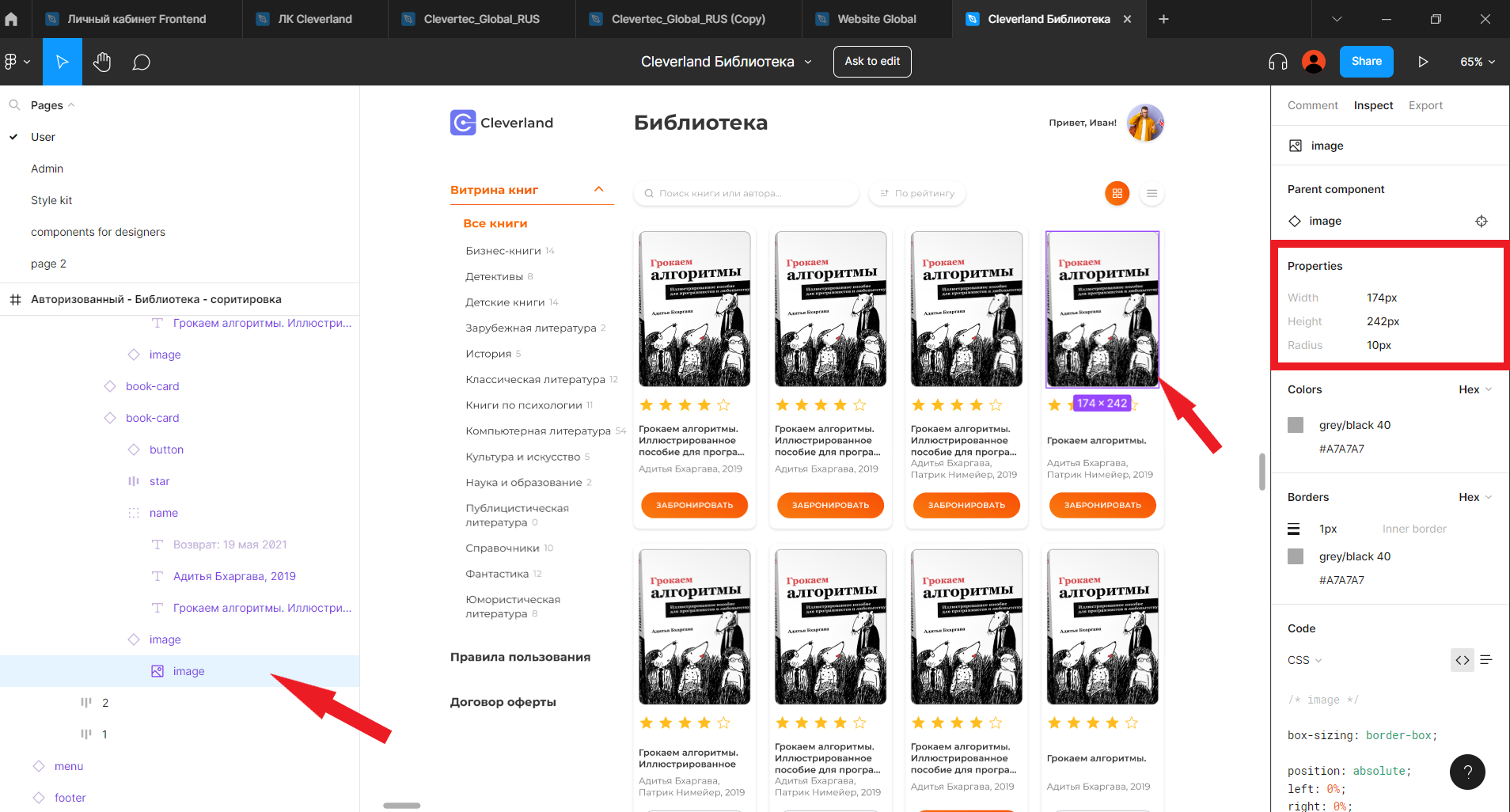
### 

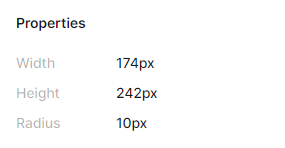
### 

### 

### **Изображение**

Свойства width и height во вкладке Inspect расскажут о размере изображения.

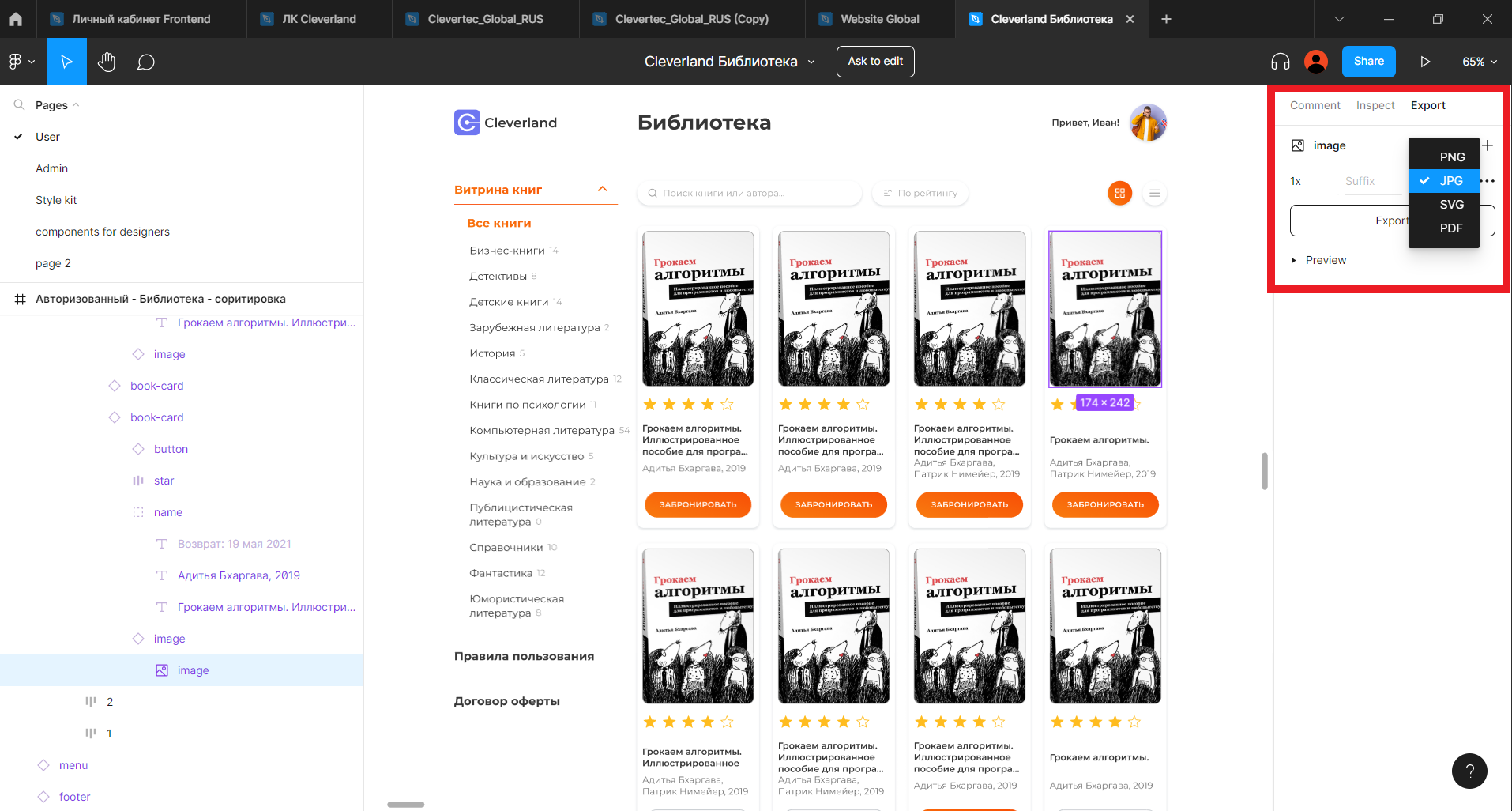




Работа с изображениями

Также можно скачать любое изображение из макета. Для этого нужно кликнуть по вкладке Export и там найти последний пункт Export. Далее в выпадающем меню выбрать формат, в котором вы хотите экспортировать изображение, — PNG, JPG, SVG или PDF.

Подробно о форматах изображений можно почитать в [этой статье](https://www.calltouch.ru/blog/formaty-graficheskih-izobrazhenij-kakie-byvayut-i-dlya-chego-ispolzuyutsya/).

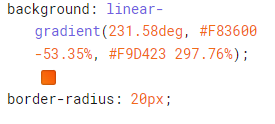


Экспорт изображений

### **Цвет элемента**

Во вкладке Inspect можно узнать фоновый цвет элемента — значение background.





Определение фонового цвета блока

### **Расстояние между элементами**

Можно узнать расстояние между любым элементом и соседними с ним. Для этого нужно выделить его, а затем наводить курсор на остальные элементы — будет появляться направляющая и значение.



Определение размеров элементов и расстояния между ними.

**Результативно выполненным заданием считается:**

* сверстанные страницы при помощи библиотеки **React**:

1. главная страница
2. страницы “Правила пользования” и “Договор оферты”
3. страница книги

* Переход между этими страницами должен осуществляться при помощи библиотеки **React Router.**
* При вёрстке необходимо использовать основные принципы [адаптивности](https://html5book.ru/adaptivnaya-vyorstka-sayta/) и [семантики](https://html5css.ru/html/html5_semantic_elements.php), также верстка на ключевых разрешениях (1440, 768, 320 px) должна совпадать с макетом.

Различные эффекты (бургер-меню, слайдеры и т.д.) и логические действия (фильтрация, поиск книг, оставление комментария, и т.д.) не нужно делать в данном спринте. Чистая сухая вёрстка.

***БИБЛИОТЕКА REACT***

**React — это JavaScript-библиотека для создания пользовательских интерфейсов.**

[Документация по React](https://ru.reactjs.org/docs/getting-started.html)

React использует виртуальный [DOM](https://ru.wikipedia.org/wiki/Document_Object_Model) (*virtual DOM*). React создаёт [кэш](https://ru.wikipedia.org/wiki/%D0%9A%D1%8D%D1%88)-структуру в памяти, что позволяет вычислять разницу между предыдущим и текущим состояниями интерфейса для оптимального обновления DOM-браузера. Таким образом программист может работать со страницей, считая, что она обновляется вся, но библиотека самостоятельно решает, какие компоненты страницы необходимо обновить.

React поддерживает специальный язык разметки JSX. Он представляет собой смесь синтаксиса HTML и JavaScript, которая выглядит примерно следующим образом:



JSX — это абстракция, которая позволяет использовать синтаксис HTML внутри вашего кода [JavaScript](https://www.digitalocean.com/community/tutorial_series/how-to-code-in-javascript) и с помощью которой вы можете создавать компоненты React, которые выглядят как стандартная HTML-разметка. JSX — это язык шаблонов для элементов [React](https://reactjs.org/), поэтому он служит основой для любой разметки, которую React будет отображать в вашем приложении.

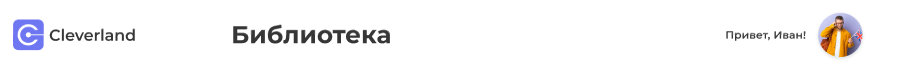
React — это инструмент для создания пользовательских интерфейсов. Его главная задача — обеспечение вывода на экран того, что можно видеть на веб-страницах. React значительно облегчает создание интерфейсов благодаря разбиению каждой страницы на небольшие фрагменты. Эти фрагменты называются компонентами.

***ОБРАТИТЕ ВНИМАНИЕ плюсом будет использование TypeScript*** или проверка типов с помощью **PropTypes**.

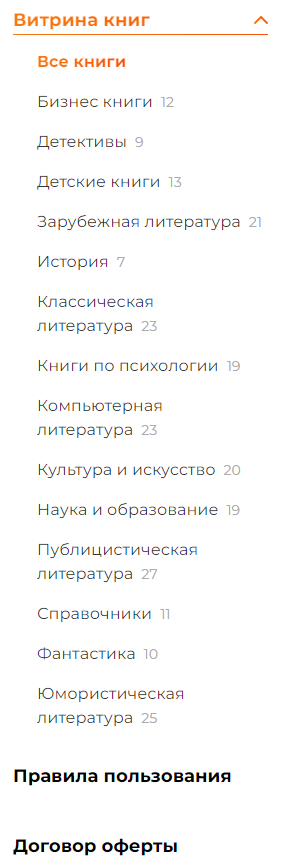
Валидация свойств с помощью React PropTypes — полезный способ принудительного использования компонентов. Это поможет избежать во время разработки будущих ошибок и проблем, когда приложение станет больше. Это также делает код более читаемым, так как мы можем видеть, как использовать каждый компонент.

Вот пример разбивки страницы на компоненты нашего проекта:

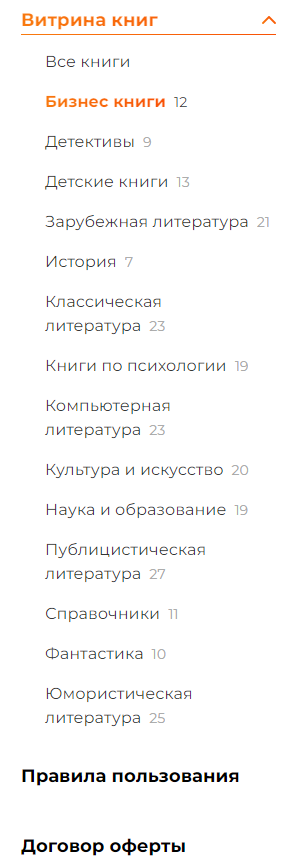
**Header сайта:**



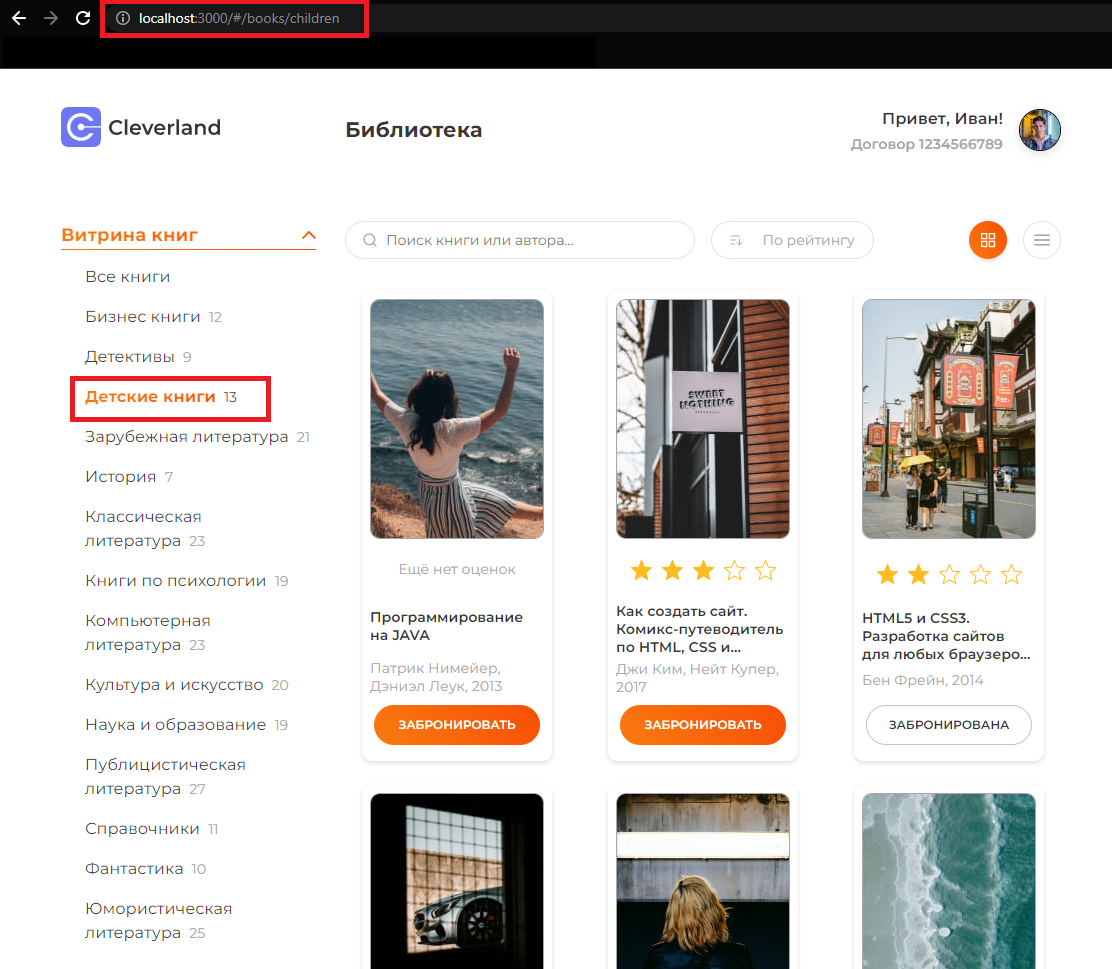
**Навигационное меню:**



**Навигационное меню (выбран определенный жанр):**



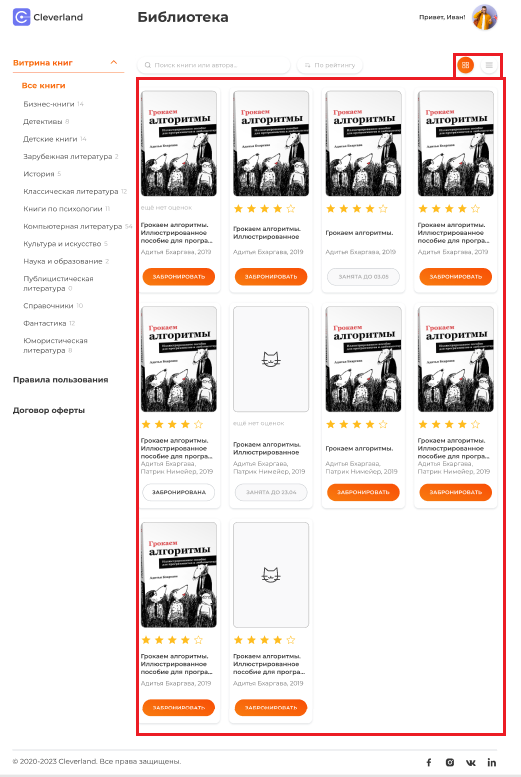
***ОБРАТИТЕ ВНИМАНИЕ При выборе жанра также должен меняться роут.***



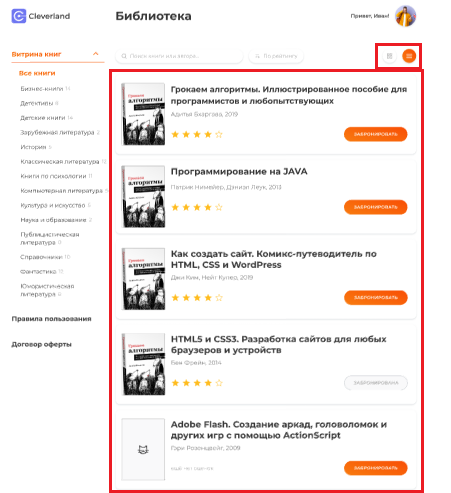
**Меню фильтрации, поиска, переключения вида отображения книг:**



**Контентная часть в виде плитки:**



**Контентная часть в виде списка:**

****

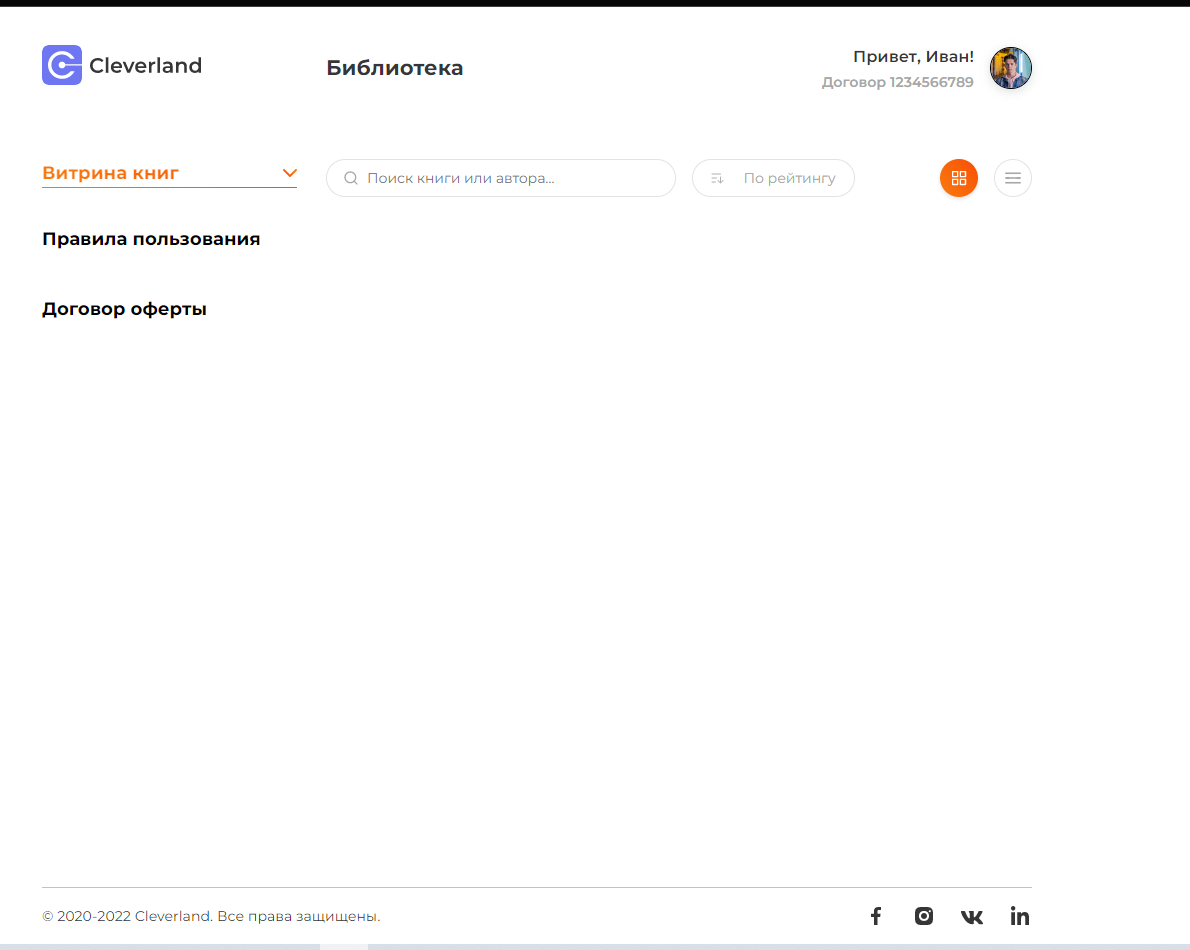
***Обратите внимание на дизайн активной кнопки при различном отображении книг в виде списка и плиток!***

***Обратите внимание на вид книг, в которых отсутствует обложка (должна быть заглушка – иконка котика)***

**Footer сайта:**

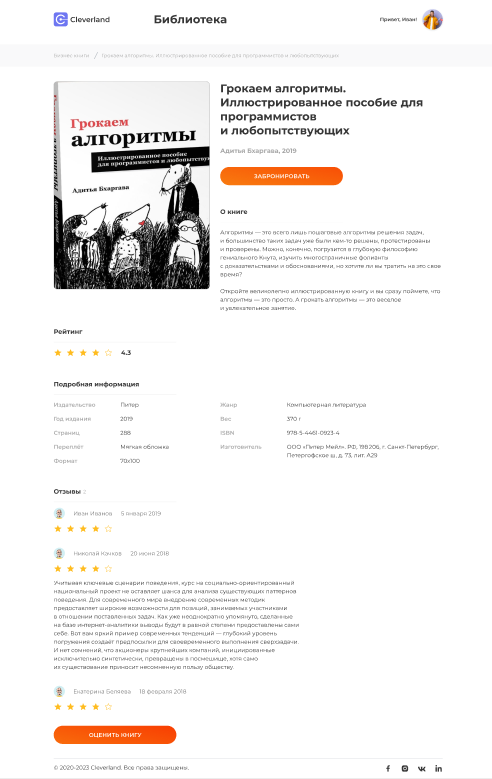


***Обратите внимание, что при отсутствии контента Footer необходимо прижать к нижней части экрана:***



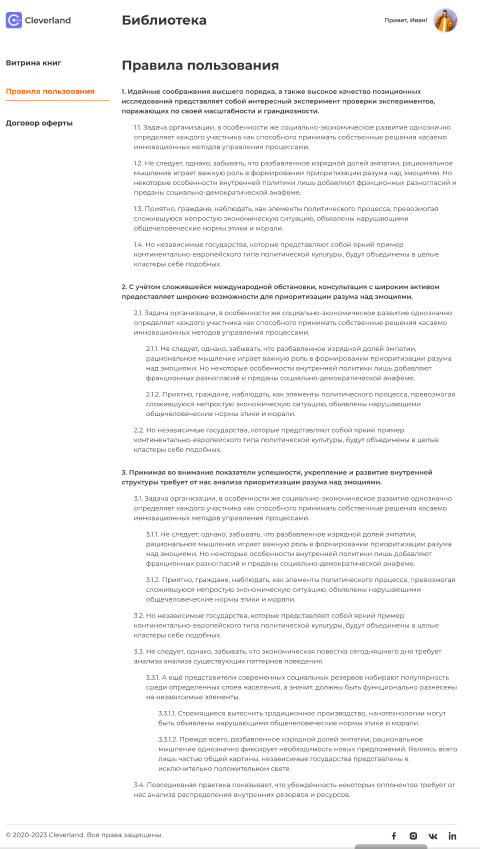
Глядя на макет Figma, можно сделать вывод, что Header и Footer в проекте будут использоваться на всех страницах, которые надо отверстать. И это наглядный пример переиспользования компонентов.

**Страница книги:**



***Обратите внимание, что при отсутствии обложки должна быть заглушка – иконка котика.***

**Страницы “Правила пользования” и “Договор оферты”**



# ***REACT ROUTER***

**React Router —** это стандартная библиотека маршрутизации (routing) в React. Он хранит интерфейс приложения синхронизированным с URL в браузере. React Router позволяет вам маршрутизировать "поток данных" (data flow) в вашем приложении понятным способом. Он подобен утверждению, если у вас есть данный URL, он будет подобен этому Route (маршруту), и интерфейс будет таким!

Правильный роутинг ссылок и загрузка разных страниц в зависимости от параметров заданных маршрутов — это главное требование для буквально любого приложения.

В данном проекте необходимо использовать React Router V6.

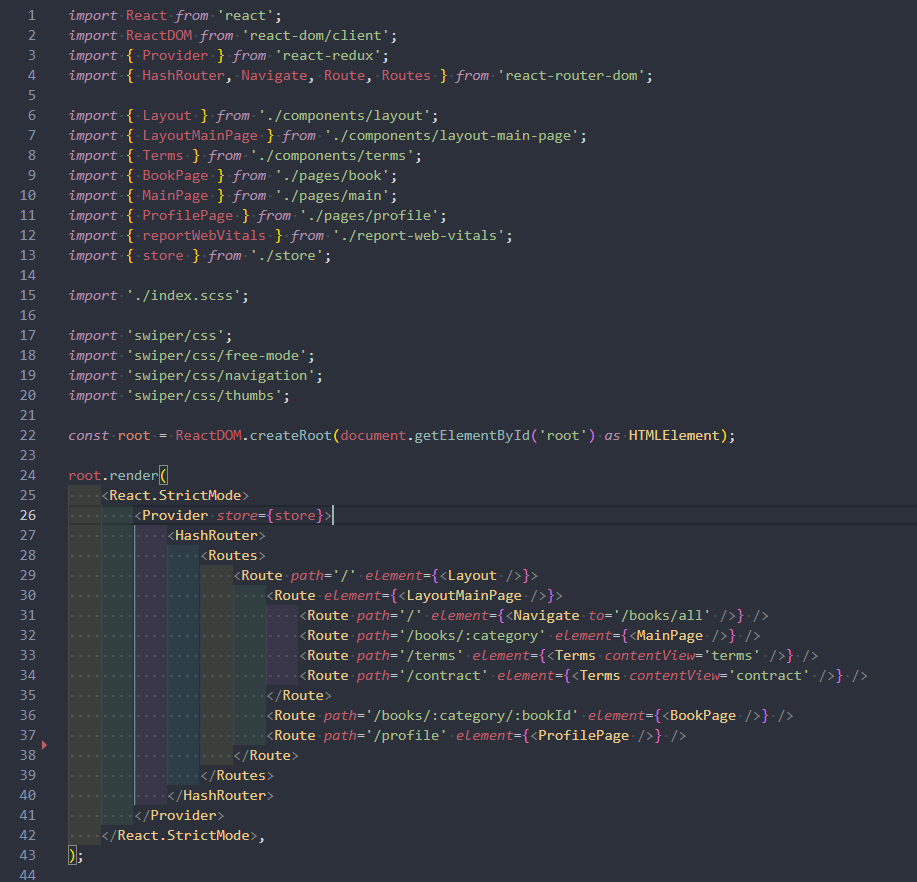
Документацию можно найти на [официальном сайте](https://reactrouter.com/)

Перед тем как углубиться в код роутера, давайте поймем, что же нам надо для роутинга страниц в нашем приложении.

1. Ссылки для навигации между страницами.
2. Нам нужно будет определить пути к страницам. Тут мы укажем путь URL и компонент, который нужно загружать с этим URL.
3. Ещё мы определим роутер, который будет проверять наличие запрашиваемого URL в указанных роутах.

Для начала необходимо установить пакет yarn add react-router-dom или npm i react-router-dom.

*Дальнейший действия разберем на ПРИМЕРЕ*.



Давайте пройдем по каждой строке отдельно.



Тут мы импортируем следующие компоненты:

* **Navigate**, при переходе на роут, указанный в path, происходит перенаправление на роут в Navigate.
* **Route**, который содержит маршруты и компоненты.
* **Routes**,в котором будет храниться вся логика нашего роутинга. Когда пользователь кликает на ссылку, этот компонент проверяет, есть ли она в указанных роутах. Если да, то роутер сменит URL в браузере и роут отрендерит заданный компонент.
* **HashRouter** - это подвид маршрутизатора с использованием символа # в URL для синхронизации пользовательского интерфейса: как пример — ссылка вида [http://localhost:3000/#/](http://localhost:3000/#/about)books. Hash-маршрутизатор применяется на статических веб-сайтах.  
  **HASHROUTER НЕОБХОДИМО ИСПОЛЬЗОВАТЬ ОБЯЗАТЕЛЬНО!!!**

**Routes** должен быть родительским компонентом для Link и Route. Собственно так он и управляет роутингом. Если мы поставим Link или Route за его пределы, то мы получим ошибку.

Зачем нам нужен Link, а не HTML анкор с href?

HTML-тег a создает ссылку на сервер. Так что каждый раз, как пользователь кликает на роут, он не будет проверять наличие роутера в роуте. Вместо этого он просто редиректнет страницу в браузере на этот роут.

В то время как Link проверяет роутер и роутер проверяет наличие роута, загружая компонент без перезагрузки страницы. Поэтому это и называется client side роутингом. Он не подгружает страницу с сервера при обращении к компоненту Link.



Тут у **Route** есть путь и props-компоненты. **element** помогает отрендерить компонент, когда пользователь заходит на этот роут. **path** props указывает путь url, который должен совпадать при посещении пользователем страницы.

Теперь обратимся к проекту.

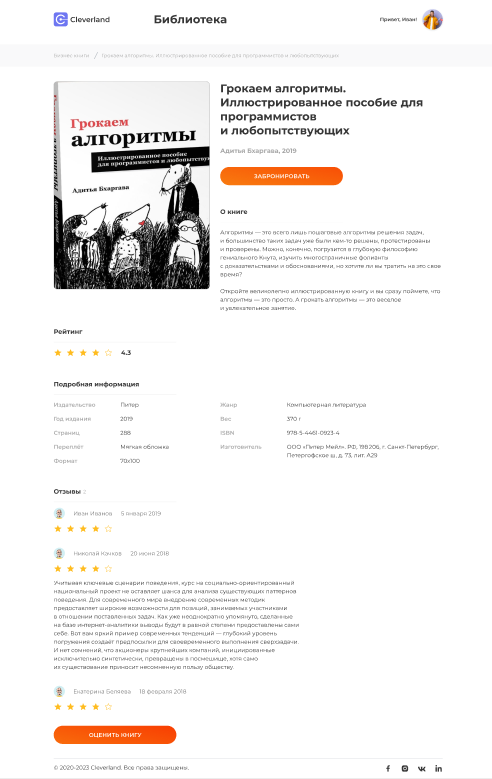
Ссылка на главную страницу:



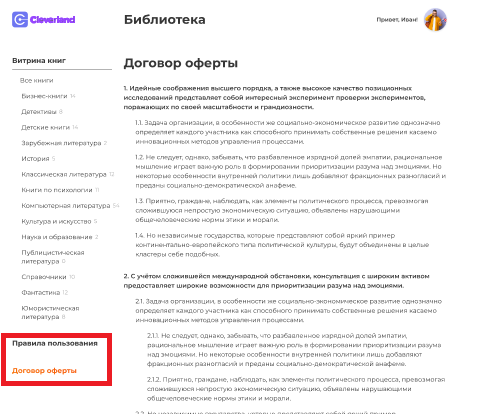
Ссылка на страницу книги (при нажатии на карточку книги):



Вот на эту страницу:



При нажатии на данные кнопки происходит переход на соответствующие страницы “Правила пользования” и “Договор оферты”



***СТИЛИЗАЦИЯ ПРИЛОЖЕНИЯ REACT***

Стилизовать необходимо, используя модули CSS, SCSS, или PostCSS.

***ОБРАТИТЕ ВНИМАНИЕ, что пользоваться UI библиотеками (React Bootstrap, Material-UI и им подобными) запрещается!***

***И ЕЩЕ НЕМНОГО ТЕХНИЧЕСКИХ ВОПРОСОВ***

Для успешного прохождения автотестов необходимо некоторые компоненты снабдить атрибутом data-test-id. Автотест сделает скриншот этого компонента и сравнение с оригиналом в Figma.

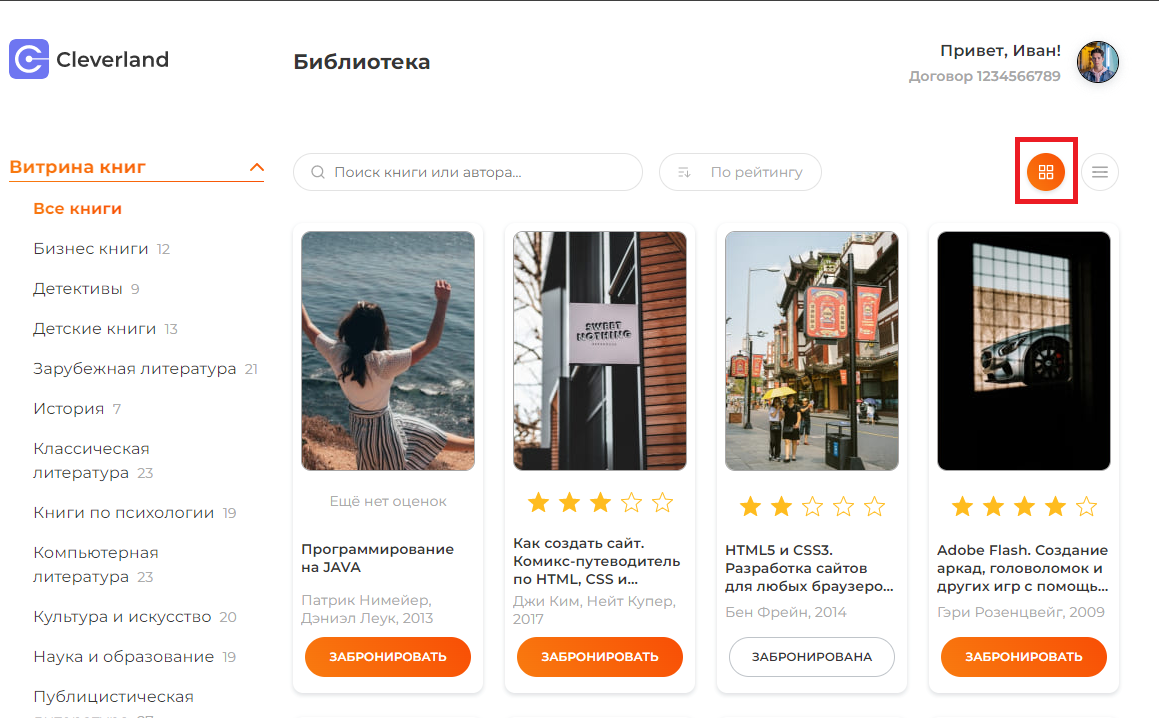
Пример использования data-test-id:

Пример data-test-id и компоненты, к которым их нужно применить:

Для компонента App данный атрибут уже применен (вы можете увидеть его в файле “publick/index.html”): *data-test-id*='app';

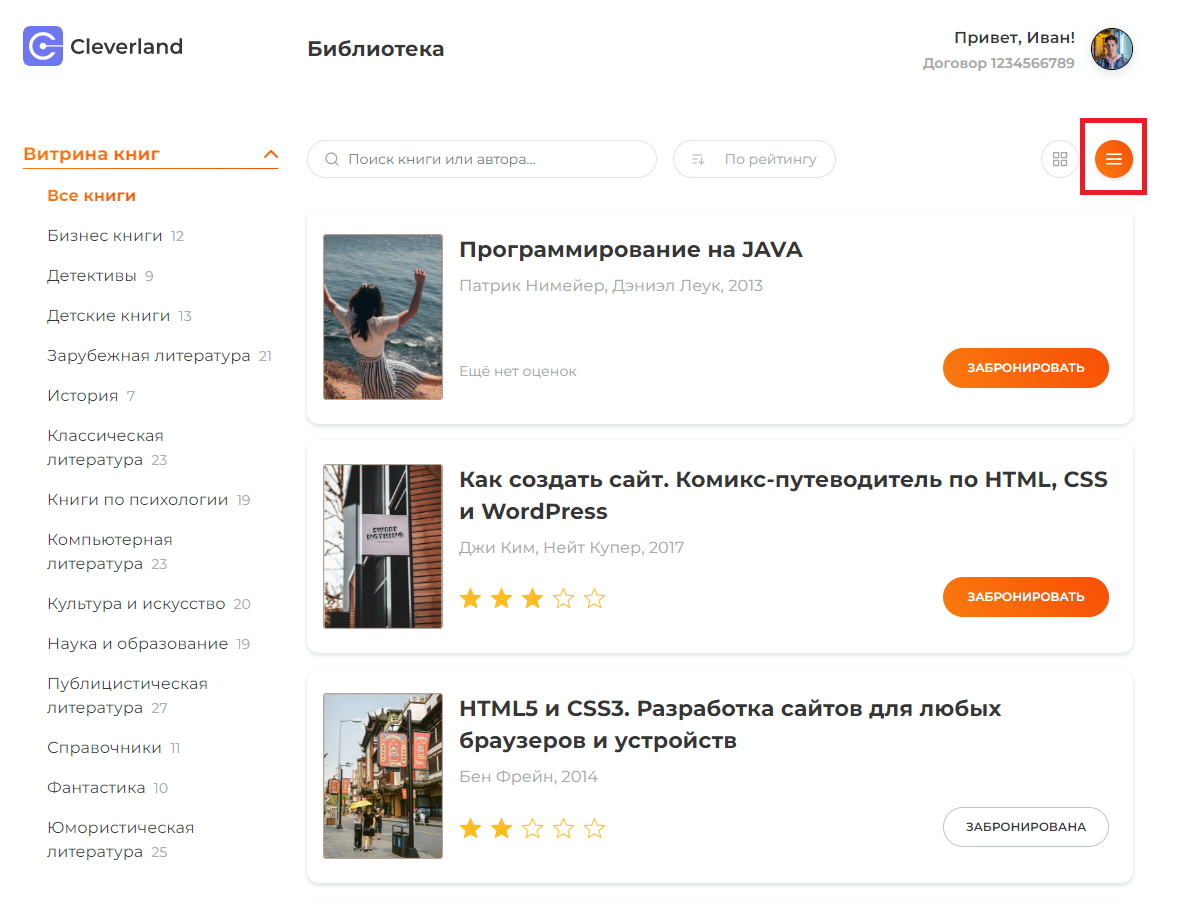
Для кнопки отображения карточек в виде окон:

*data-test-id*='button-menu-view-window';



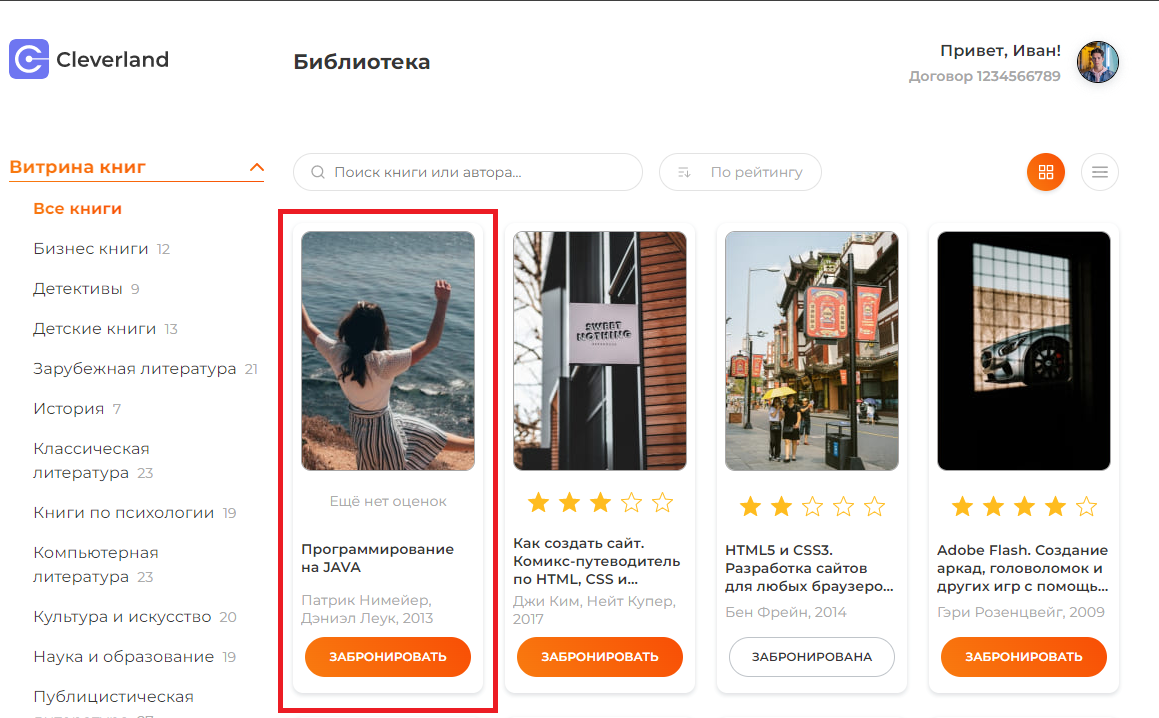
Для кнопки отображения карточек в виде списка:

*data-test-id*='button-menu-view-list';



Для компонента Card (карточка книги):

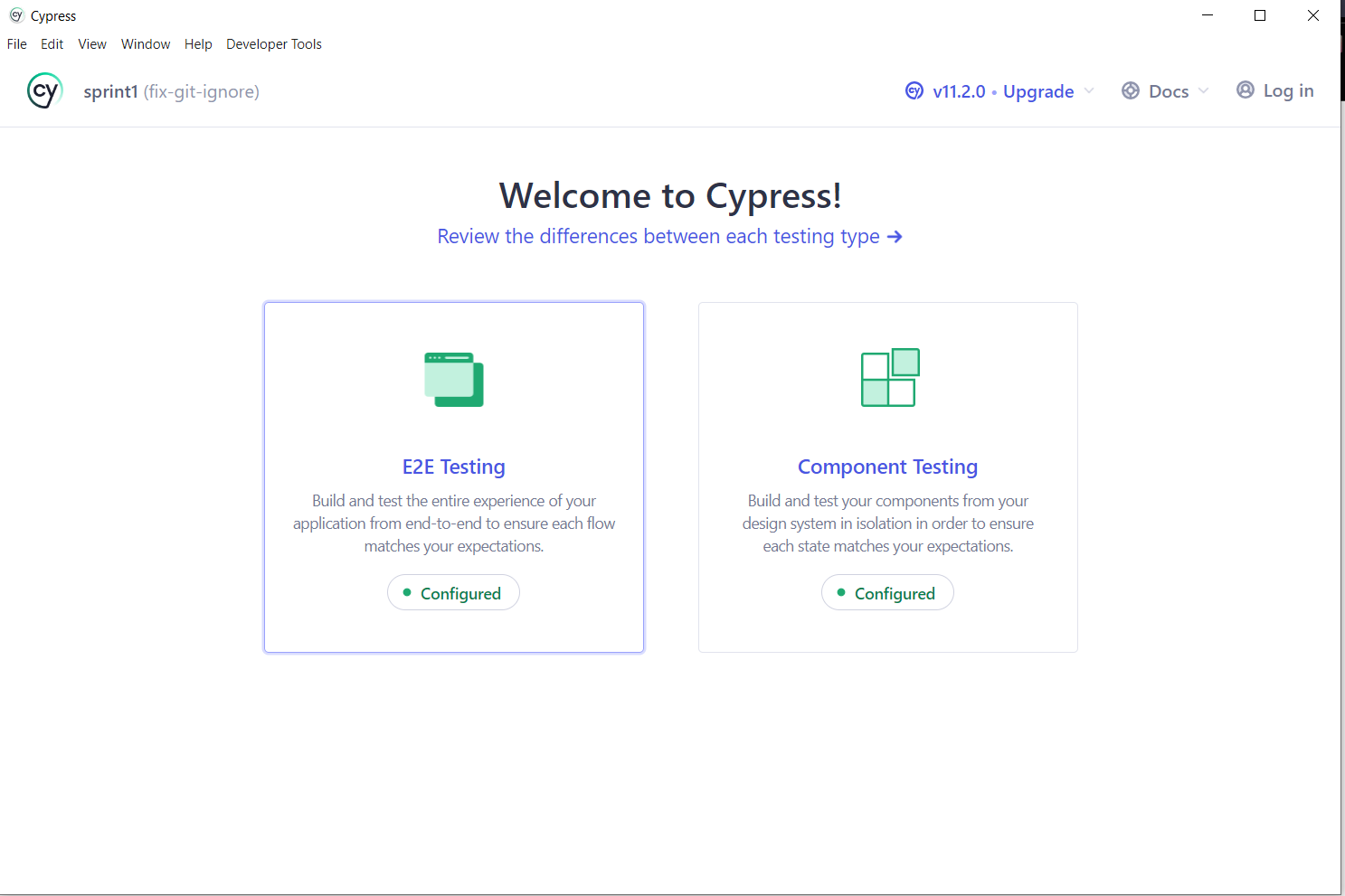
*data-test-id*='card'



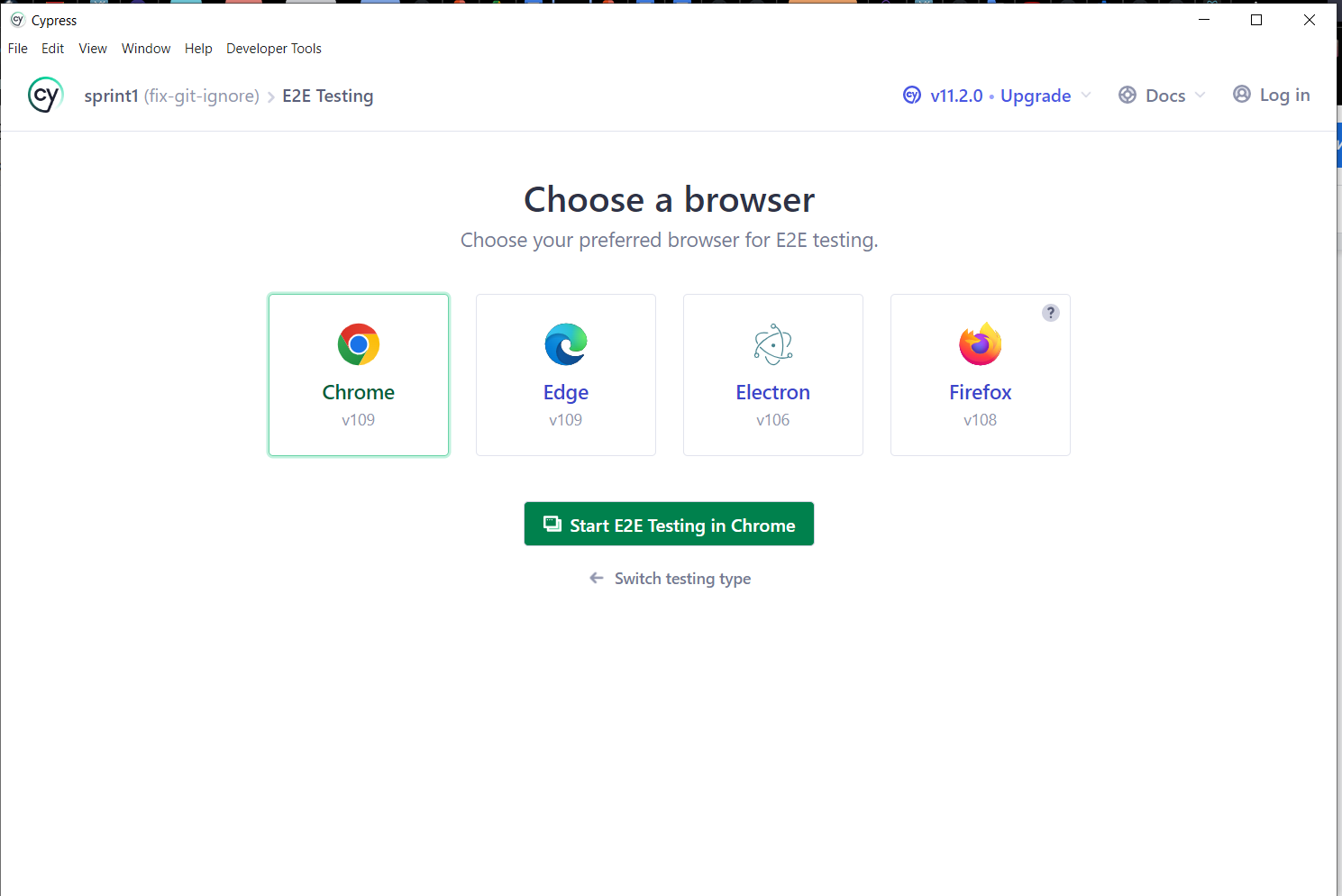
Прежде чем отправлять вашу работу на проверку, убедитесь что вы проходите минимальный процент проверки автотестами (в данном спринте – 80%). Это можно сделать, запустив локально тест на текущий 1 спринт.

Ниже краткая инструкция, как это сделать:

1. Запустите скрипт “е2е” (например npm e2e)
2. В появившемся окне выберите Е2Е Testing



1. Выбираете браузер и стартуете тесты



1. Выбираете текущий спринт



1. Тесты запускаются автоматически (при изменении вашего кода также произойдет перезапуск тестов)
2. Советуем периодически чистить папку /cypress/report

Подведем итоги! В конце первого спринта у вас в портфолио будет сверстана ***1. Главная страница***, содержащая:

* Хедер
* Навигационное меню, включающее выбор жанра книг, ссылки на страницы “Правила пользования” и “Договор оферты”
* Меню с инпутом поиска книги и кнопками фильтрации, смены вида контента
* Контент, изменяющийся в зависимости от выбранного вида отображения
* Футер

**2. *Страница книги*.  
*3. Страницы правил пользования и договора оферты***

**В данном спринте необходимо реализовать только верстку, переход на страницу книги при клике на ее карточку и смену вида контента на главной странице. Все размеры, шрифты, отступы, эффекты наведения должны соответствовать макету.   
 Не нужна логика отображения бургер-меню, слайдеров, смена вида поиска и т.д.**

**На данный спринт отводится 2 недели – начало спринта №1 19.01.2023 окончание 02.02.2023 12:00**

Всем удачи!!! Да прибудет с Вами Сила!!!Смайлики из Звездных Войн  
P.S. По вопросам или за разъяснениями обращайтесь в телеграм-канал.