МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ  
Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ   
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 1-40 05 01 «Информационные системы и технологии»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА

по дисциплине «Скриптовые языки программирования»

Тема «Web-сайт чайного магазина»

Исполнитель

студент(ка) 2 курса 1 группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е. В. Быско

подпись, дата

Руководитель

доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.А. Жиляк

должность, учен. степень, ученое звание подпись, дата

Допущен(а) к защите \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата, подпись

Курсовой проект защищен с оценкой

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.А. Жиляк

подпись дата инициалы и фамилия

Содержание

[1. Обзор технических методов и программных средств разработки веб-сайта 4](#_Toc121710655)

[1.1 Обследование объекта и обоснование необходимости создания ПП. 4](#_Toc121710656)

[1.2 Формирование требований к ПП. 4](#_Toc121710657)

[1.3 Назначение и цели создания системы. 5](#_Toc121710658)

[1.4 Требования к системе. 5](#_Toc121710659)

[1.5 Поставновка задачи. 6](#_Toc121710660)

[2. Последовательность разработки содержания и структуры веб-сайта с использованием компонентов 7](#_Toc121710661)

[2.1 Техническое задание. 7](#_Toc121710662)

[2.2 Построение прототипов интерфейсов и макетирование страниц. 7](#_Toc121710663)

[2.3 Разработка дизайна элементов ПП. 7](#_Toc121710664)

[2.4 Составляющие структуры программного продукта. 8](#_Toc121710665)

[2.5 Вывод. 9](#_Toc121710666)

[3. Программная реализация проекта 10](#_Toc121710667)

[3.1 Вёрстка программного продукта. 10](#_Toc121710668)

[3.2 Вывод. 11](#_Toc121710669)

[4. Ключевые функции разработанного программного кода, создание значимых документов 12](#_Toc121710670)

[4.1 Руководство пользователя. 12](#_Toc121710671)

[4.2 Кроссбраузерное тестирование. 12](#_Toc121710672)

[4.3 Тестирование на адаптивность. 13](#_Toc121710673)

[4.4 Вывод. 14](#_Toc121710674)

[5 Заключение. 15](#_Toc121710675)

[6 Список использованной литературы. 16](#_Toc121710676)

[Приложение. 17](#_Toc121710677)

[7.1 Приложение 1. 17](#_Toc121710678)

[Приложение 2 18](#_Toc121710679)

ВВЕДЕНИЕ

Присутствие бизнеса в Интернете оказывает большое влияние на его успех. В большинстве случаев клиенты перед совершением покупки обращают внимание на наличие веб-сайта. Его наличие присутствия в Интернете, может иметь решающее значение для бизнеса. Кроме всего этого веб-сайт является лицом бренда и компании в общем.

Бизнес должен иметь свой собственный веб-сайт по целому ряду причин. Одна из основных — демонстрация основного ассортимента товаров потенциальным клиентам. Также это помогает чётко определить, что из себя представляет компания, и повысить свои шансы на успешную продажу своего продукта клиентам.

Веб-сайт — то, что помогает выделить компанию среди конкурентов. Благодаря ему можно найти покупателей, которые заинтересованы именно в данной продукции, показать свои лучшие товары или сильные стороны. Кроме этого можно продвинуть свою новую продукцию или услугу. Также заинтересованные люди смогут связаться с компанией напрямую с помощью информации, указанной на веб-сайте, что также увеличит продажи. Поскольку веб-сайт работает круглосуточно и без выходных, то компания может легко публиковать обновления и объявления для своих клиентов, а также быстро обновлять информацию, если это будет необходимо.

Веб-сайт даёт возможность использовать различные приёмы интернет-маркетинга: уведомления об акциях, реклама, создание всплывающих окон и т.п.

Данные факты отражают актуальность тематики курсовой работы.

Целью курсового проекта являетсяразработать веб-сайт для чайного магазина с использованием HTML5, а также с применением React.js.

К задачам курсового проекта относится:

1. Проанализировать дизайн существующих веб-сайтов, инструменты и библиотеки для создания веб-сайта, а также его оформления.

2. Разработать макет и прототип сайта.

3. Разработать структуру веб-сайта.

4. Наполнить веб-сайт информацией, соответствующей определённой теме.

5. Создать дизайн сайта, используя React.js, а также создать интерактивные элементы

5. Протестировать веб-сайт.

Целевой аудиторией будут люди, которые хотят узнать о самых ярких представителях разных видов чая, их правильном заваривании, а также различных добавках, которые сочетаются с определённым видом чая или богаты витаминами и улучшают вкус.

Также одной из целевой аудиторий являются люди, которые хотят заказать какой-нибудь из видов чая или же добавок к нему.

# 1. Обзор технических методов и программных средств разработки веб-сайта

# 1.1 Обследование объекта и обоснование необходимости создания ПП.

На данный момент существует огромное количество способов продвижения своей продукции в Интернете. Некоторые создают красивые сайты, которые легко бросаются в глаза и быстро запоминаются посетителям, другие же – могут похвастаться отличным наполнением своего веб-сайта, третьи – обладают отличным качеством продукции, поэтому в погоне за таким товаром покупатели возвращаются вновь после покупки какого-то товара из ассортимента.

Однако вне зависимости от того, чем может похвастаться кампания, которая продаёт свой товар в сети Интернет, ей нужен веб-сайт. Когда посетитель видит какой-то магазин или ищет какую-то определённую продукцию, он в первую очередь пользуется Интернетом и начинает искать те магазины, которые подойдут именно ему по различным критериям, например, цена или месторасположение магазина. В таких случаях веб-сайт данной кампании является главным лицом от которого зависит купит ли в итоге покупатель именно в этой кампании товар или нет. На исход влияет огромное количество факторов начиная от эстетичности страниц, количества информации о кампании и самих товарах, заканчивая отзывами на других сайтах, чтобы удостовериться в уровне обслуживания в этой кампании.

Именно учитывая всё вышеперечисленное можно понять, что веб-сайт – важная составляющая каждого бизнеса, от которой в некотором смысле будет зависеть обанкротиться ли кампания или станет популярной через время.

# 1.2 Формирование требований к ПП.

В данной курсовой работе веб-сайт создавался с помощью фреймворка React, также использовался Scss для задания стилей. Данный фреймворк удобен тем, что есть возможности использовать скрипты, которые позволяют, например, динамически обновлять страницу, также React имеет своё встроенное DOM-дерево, что позволяет без особых трудов сократить нагрузку на компьютер клиента, который будет посещать данный веб-сайт, это достигается путём того, что React отслеживает изменившиеся компоненты и обновляет только их а не всю страницу целиком, как это происходит в обычных HTML-страницах.

Основными требованиями к данной курсовой работе являются:

Лаконичность и простота дизайна, страницы не должны быть сильно перегружены информацией, но при этом должны отражать всю суть данной страницы и самого проекта в целом. Это достигается путём добавления только ключевой, значимой для посетителя и потенциального покупателя информации. В описаниях и на страницах веб-сайта также не должно быть излишней информации, которая усложнит поиск предполагаемо интересуемой информации посетителя веб-сайта.

Благодаря простоте дизайна есть возможность использовать базовую (основную) палитру цветов, выбранную для этого веб-сайта, а также визуально обратить внимание посетителя именно на те части веб-сайта, которые необходимо выделить, например, для повышения количества продаж или других целей.

Текст на всех страницах вне зависимости от его расположения должен иметь оптимальный размер, а также его можно было бы без затруднений прочитать. Благодаря такому тексту посетителю не надо будет долго всматриваться в текст, в попытках разобраться, что там написано, а, следовательно, это не отпугнёт потенциальных покупателей и, возможно, повысит продажи.

Навигация на веб-сайте должна быть понятна при беглом просмотре глазами. Это поможет посетителю веб-сайта не «заблудиться» во всех страницах, которые содержит посещённый им веб-сайт, а также быстро переключаться между нужными страницами.

Кроме всего эта навигация должна быть удобной. Следует избегать близкого расположения ссылок на другие страницы, так как пользователь может случайно промахнуться и, как итог, перейти на другую страницу, чего пользователю явно не хотелось.

Особое внимание стоит уделить адаптивности веб-сайта. Благодаря адаптивности веб-сайт корректно отобразиться на всех устройствах, а также размерах экрана. Как итог сайт будет выглядеть презентабельно и красиво на каком бы устройстве его не открыли. Также благодаря этого это расположит клиента, и он с большей вероятностью может приобрести продукцию.

# 1.3 Назначение и цели создания системы.

Назначение данной системы является разработка корректно работающего приложения с использованием фреймворка React, а также использование Scss для задания стилей.

Целями же является создание своих компонентов, которые будут корректно функционировать в приложении, а также использование готовых компонентов для получения наилучшего результата. Также одной из целей является создание работающего проекта, который содержит больше одной веб-страницы.

# 1.4 Требования к системе.

Основными требования к системе являются: корректная работа всего сервиса, стабильность при работе с данной системой, также сюда входит отказоустойчивость. Кроме всего прочего система не должна «тормозить» при выполнении пользователем действий, или при посещении сайта, или при использовании функционала сайта. Кроме всего прочего компоненты, которые составляют данную систему, должны иметь возможность переноситься в неизменном виде и на другие сервисы, если такое необходимо.

Требованиями к функциям, выполняемым системой, являются: правильность работы при их выполнении, также функции не должны сложно читаться. Кроме всего этого функции должны быть отказоустойчивыми и разделены на несколько файлов. Данные файлы создаются в зависимости от таких критериев как, логика самого приложения, результат выполнения функции. Данные функции должны иметь возможность импортироваться для использования в других местах приложения. Основные функции размещаются в тех же файлах, где и их вызов для простоты чтения кода и отсутствия загруженности огромным количеством файлов, что может привести к затруднению понятия логики приложения.

Основными требованиями к видам обеспечения являются: наличие установленного фреймворка React. Также необходимо подключение к сети Интернет и любой установленный браузер. Данная система может работать на всех операционных системах с графической оболочкой подобной Windows. Кроме всего прочего в корневой каталог необходимо доустановить модули ReactRouter и ReactCart.

# 1.5 Поставновка задачи.

Данный курсовой проект представляет собой приложение, которое написано с использованием фреймворка React, а также Scss для задания всем элементам на странице стилей. Проект является многостраничным и написан по теме «Чайный магазин». В проекте реализованы функции для корректной работы сортировки, поиска, перехода между страницами и т. д. Кроме всего прочего данный проект будет содержать компоненты, которые можно переносить на другие сервисы с минимальными изменениями в логике их работы.

# 2. Последовательность разработки содержания и структуры веб-сайта с использованием компонентов

# 2.1 Техническое задание.

В данном курсовом проекте необходимо создать веб-сайт Чайного магазина с использованием фреймворка React, а также использовать Scss для задания стилей всем элементам на странице. React является основой данного проекта, который создаёт основное приложение с помощью create-react-app. Сам проект состоит из 4 страниц: главная, ассортимент с товаром, корзина, страница о кампании.

Главной задачей веб-сайта является создание многостраничного проекта используя функции, встроенные и пользовательские хуки, а также дополнительные компоненты для поддержания корректной работы приложения. Кроме всего прочего все страницы должны быть адаптивны и корректно работать.

Все страницы имеют одинаковую верхнюю и нижнюю части. На верхней части расположены ссылки на все остальные страницы приложения, а внизу расположена общая информация, контакты и также ссылки на другие страницы.

Наполнение страниц соответствует их названию.

Данный проект, а также дизайн является адаптивным и кроссбраузерным (корректное отображение в таких браузерах, как Google Chrome, Mozilla Firefox, Microsoft Edge).

# 2.2 Построение прототипов интерфейсов и макетирование страниц.

Для большей наглядности и простоты исправления ошибок проектирования и дизайна было выбрано приложение Figma. Данное приложение позволило создать прототип всех страниц сайта и понять, как будет выглядеть сайт в конечном итоге. Также использование функции соединения компонента на одной странице с компонентом на другой странице или же целой страницей позволяет в режиме просмотра сделать прототип функционирующего многостраничного сайта. Прототипы страниц можно посмотреть в Приложении 1.

# 2.3 Разработка дизайна элементов ПП.

Для достижения адаптивности использовалась вёрстка с использованием flex-элементов. Данная вёрстка помогает легко расположить элементы в нужном порядке, а также делает элементы адаптивными и подстраиваемыми под любой размер экрана.

Все стили задаются с помощью Scss, который с помощью Ruby автоматически перекомпилирует файл в css, который можно использовать для задания стилей элементам. Весь сайт сделан с использованием белого, чёрного и различных оттенков серого цветов. Такая цветовая палитра не является слишком броской, легко может сочетаться с другими цветами для обращения внимания клиента на какие-либо элементы, также данный дизайн смотрится довольно лаконично и не нагружает страницы веб-сайта.

Рисунок 2.1 – Часть веб-страницы с ассортиментом магазина

Также используется большое количество картинок, которые отображают ассортимент магазина. Благодаря и можно увидеть какой товар продаётся, а также это является показателем того, какой товар получит покупатель при заказе.

Также ко всем элементам сайта на всех страницах применяется пользовательский шрифт, который имеет название Marga.

# 2.4 Составляющие структуры программного продукта.

Приложение созданное с помощью create-react-app состоит из нескольких папок. Все основные файлы находятся в папке src. В этой папке находятся файлы со всеми страницами, которые используются в данном проекте.

В папке items находятся файлы, которые выводят на страницу, например, ассортимент товаров (tableBody, tableHeader и другие), поиск товара (Search), сортировка по различным критериям (SortTable), а также массив, который содержит всю информацию о товаре (Content).

В папке photo содержатся все фотографии, которые использу.тся в данном проекте.

В папке style находятся все стили, которые используются в данном проекте. Там папки делятся на ещё несколько. Папка css используется при компиляции Scss в css. Из этой папки следует подключать стили. В папке scss находятся сами файлы scss, которые используются, если необходимо задать или изменить стили какому-либо из элементов на текущей странице или любой другой.

# 2.5 Вывод.

В данном разделе была определена последовательность разработки веб-сайта. Описан дизайн элементов, создание макетов в виде прототипа проекта. А также составлена структура программного продукта.

# 3. Программная реализация проекта

# 3.1 Вёрстка программного продукта.

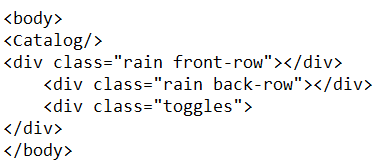
Так как весь проект реализовывается с помощью приложения create-react-app, то обычный HTML использовать нельзя. Весь код, который необходимо вывести в виде страницы, расположен в функции или компоненте, который потом указывается в необходимом месте, а после этого корректно отрисовывается при запуске его в браузере.

Рисунок 3.1 – Часть кода, где Catalog – импортированный элемент

Для использования функций не только в данном файле, но и в любых других, которые не выходят за пределы директории src, необходимо первым слово при создании функции указывать export. При это для добавления функции в какой-либо файл необходимо писать слово import.

Для создания много страничного файла использовался React Router. В нём есть элементы, которые можно экспортировать и тем самым создать многостраничный веб-сайт.

|  |  |
| --- | --- |
| а | б |

Рисунок 3.2 – Часть кода а – в файле App.js, б – которая показывает как используется переход между страницами вне файла App.js

Для корректной работы перехода между страницами на нужных страницах используется часть кода, показанная на Рисунке 3.2(б).

Если из большого файла необходимо экспортировать только функцию, находящуюся в этом файле, то при импорте этой функции её название берут в фигурные скобки, а только после этого указывают путь к файлу, в котором лежит данная функция.

Стили подключаются также с помощью import и далее пишут путь к нужному файлу. Так как у React есть наследование стилей, то этим можно избежать избыточности кода. Например, стили из index.css файла, который создаёт автоматически при создании приложения, наследуются всеми последующими элементами в проекте.

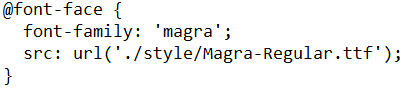
Так как в проекте используются пользовательские шрифты, то данный шрифт можно указать один раз в файле index.css, а в последующих файлах это опускать. Используются данные шрифты также как и встроенные.

Рисунок 3.3 – Подключение пользовательского шрифта

# 3.2 Вывод.

В данном разделе было описано создание проекта. Также описаны основные части его создания и нюансы, которые могут возникнуть при создании приложения с помощью create-react-app.

# 4. Ключевые функции разработанного программного кода, создание значимых документов

# 4.1 Руководство пользователя.

Для корректной работы данного разработанного React-приложения в первую очередь необходимо определиться с директорией, в которой будет находиться данный проект.

После этого необходимо в командной строке указать эту директорию и создать React-приложение. Для этого в командной строке выполняется npx create-react-app <имя\_проекта>. По прошествую времени создастся React-приложение. Необходимо перейти в директорию, которая имеет такое же название как и проект (директория <имя\_прокта), с помощью команды cd <имя директории>. Для запуска приложения необходимо написать npm start. И приложение запуститься в браузере с адресом localhost:3000/. Если этот хост занят, то будет предложено открыть на другом хосте.

Так как данный проект содержит дополнения, которые не входят в основной проект, то необходимо в корневой папке данного проекта доустановить компоненты. Как было сказано необходимо перейти в корневой каталог проекта и выполнить команду yarn add react-router-dom или npm add react-router-dom. После этого установится React Router, который используется для перехода между страницами много страничного веб-сайта. Следующей командой для установки дополнительных компонентов является npm install react-use-cart или yarn add react-use-cart. Данная команда добавит новые компоненты, которые можно использовать для создания корзины и ассортимента товаров.

Далее необходимо заменить файл в директории проекта на файл, который был скачан из этого проекта.

После всех выполненных манипуляций на странице, где запускался проект, будет главная страница данного проекта. Если же такого не происходит, скорее всего какое-то из действий выполнено неверно.

# 4.2 Кроссбраузерное тестирование.

Это тестирование, которое основано на корректном запуске приложения в разных браузерах. Благодаря этому тестированию можно понять в каких браузерах проект работает неисправно и починить это.

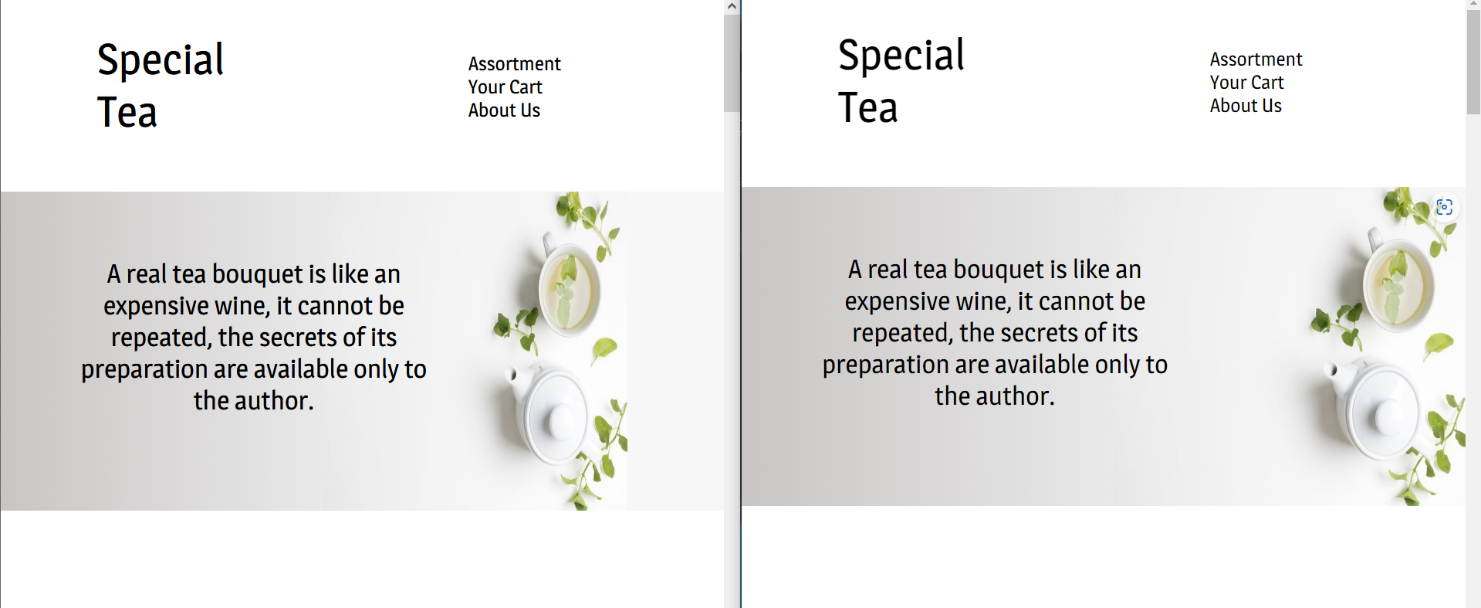


Рисунок 4.1 – слева – веб-сайт в браузере FireFox, справа – веб-сайт в браузере Microsoft Edge

# 4.3 Тестирование на адаптивность.

Согласно данному тестированию, все элементы должны корректно отображаться при различных размерах экрана, а также на различных устройствах.

Рисунок 4.2 – Пример адаптивности при размере экрана 516x915px

# 4.4 Вывод.

В данном разделе было разработано руководство пользователя, а также продемонстрировано тестирование на адаптивность и кроссбраузерность.

# 5 Заключение.

По итогам данной работы был разработан веб-сайт чайного магазина. Для его создания были использованы: create-react-app, компоненты как встроенные, так и пользовательские, а также хуки, которые помогали отслеживать и обновлять состояние на странице. Кроме этого был использован Scss для задания стилей всем элементам. Данный веб-сайт состоит из 4 страниц, которые содержат информацию о товарах, шапку, подвал, сортировку и поиск. Также благодаря Scss были реализованы кроссбраузерность и адаптивность веб-сайта, что позволяет просматривать данную работу в любом браузере и на любом современном устройстве.

Сперва были сформированы требования к программному продукту, поставлена задача, определены цели и назначения создания данного программного продукта,а также были сформированы требования к системе.

В начале работы был сделан макет сайта, а также определена основная цветовая палитра, шрифт и дизайн сайта в принципе.

Для достижения адаптивности были использованы flex-box, а также указание размеров блоков в процентах, а не пикселях или других величинах.

Были разработаны дизайны. Также описана структура программного продукта.

Следующим шагом стала вёрстка проекта, в данном разделе были разобраны только ключевые моменты и особенности при вёрстке используя create-react-app, так как от обычной вёрстки, не учитывая данных особенностей, оно не отличается.

После этого была протестирована адаптивность и кроссбраузерность. Также описана подробное руководство пользователя, которое позволит каждому запустить данный проект у себя на ноутбуке или компьютере.

# 6 Список использованной литературы.

1. <https://ru.reactjs.org/>
2. <https://www.npmjs.com/package/react-use-cart?activeTab=readme>
3. https://v5.reactrouter.com/web/guides/quick-start

# Приложение.

# 7.1 Приложение 1.

Слева – макет главной страницы, справа – макет страницы с ассортиментом

# Приложение 2

|  |  |
| --- | --- |
| App.js | Cart.js |
| import './App.css';  import Main from './Main';  import Assort from './Assort';  import Cart from './Cart';  import About from './AboutUs';  import {Routes,Route} from 'react-router-dom';  function App() {  return (  <div className="App">  <Routes>  <Route path='/' element={<Main/>}/>  <Route path='/Assort' element={<Assort/>}/>  <Route path='/Cart' element={<Cart/>}/>  <Route path='/About' element={<About/>}/>  </Routes>  </div>  );}  Catalog.js  import {SortTable} from "./items/SortTable";  import React, {useRef, useState} from "react";  import {content} from "./items//Content";  import {Search} from "./items/Search";  import './style/css/sortTable.css'  import { CartProvider, useCart } from "react-use-cart";  function Page() {  const { addItem } = useCart();  const [isShown, setIsShown] = useState(false);  const [array, setArray] = useState(content);  const [search, setSearch] = useState('');  const [sorted, setSorted] = useState({name: true, price: true, discount: true});  const [searchParameter, setSearchParameter] = useState('partial');  const [items,cart]=useState([]);  const products = content;  const sort = (byWhat) => {  let direction = sorted[byWhat] ? 1 : -1, goodsCopy = [...array].sort(function (a, b) {  if (a.new || b.new)  return 0  if (a[byWhat] > b[byWhat]) {  return direction; }  if (a[byWhat] < b[byWhat]) {  return direction \* -1; }  return 0; });  sorted[byWhat] = !sorted[byWhat];  setArray(goodsCopy); }  const sorts = () => {  for(let i=0;i<content.lengh;i++){  if(content[i].discount!=false){  content[i].push(content[i].prdis=((content[i].price \* (100 - content[i].discount)) / 100))  }else{ content[i].push(content[i].prdis=content[i].price)  }  }  return (  <div className="sorts">  <button className="btt" onClick={() => sort("name")}>  Name  </button>  <button className="btt" onClick={() => sort("price")}>  Price  </button>  <button className="btt" onClick={() => sort("discount")}>  Discount  </button>  </div>  );  const goods = () => {  return array  .filter((item) => {  if (!search) return item;  if (item.name.toLowerCase().includes(search.toLowerCase())  ) {return item; }  })  .map((item) => {  return (<>  {<div className="oneGood" key={item.id}>  <div className="wrapper">  <div className="imgDesc"  onMouseEnter={() => setIsShown(true)}  onMouseLeave={() => setIsShown(false)}>  <img className="image" src={item.img} alt=""/>  {isShown &&(<div className="dsc">{item.desc}</div>)}  </div>  <div className="text">  <div className="name">{item.name}</div>  <div className="bot">  <button className="cart" onClick={() => addItem(item)} >Cart</button>  {item.discount!=false ? <div className="prices">  <label className="dsk">{Math.ceil(((item.price \* (100 - item.discount)) / 100)\*100)/100}$</label>  <label className="lb">{item.price}$</label>  </div>: <div className="prices">  <label className="dsk1">{Math.ceil(((item.price \* (100 - item.discount)) / 100)\*100)/100}$</label>  </div>}  </div>  </div>  </div>  </div>}  </>  );  });  }  return (<>  <div className='assort'>  <div className="left">  <Search search={setSearch} searchParameter={setSearchParameter} />  <div>Type</div>  <hr/>  <ul>  <li>Red</li>  <li>Green</li>  <li>White</li>  <li>Yellow</li>  <li>Additions</li>  </ul>  </div>  <div className="assortiment">  <div className='hed'>Teas&Additions</div>  <div className="Item">  {sorts()}  <div className="goods">  {goods()}  </div>  </div>  </div>  </div>  </>  );}  export const Catalog=()=> {  return (  <CartProvider>  <Page />  </CartProvider>  );}  export default Catalog; | import {Link} from 'react-router-dom';  import './style/css/cart.css';  import { CartProvider, useCart } from "react-use-cart";  export function Cart() {  const {  isEmpty,  totalUniqueItems,  items,  updateItemQuantity,  removeItem,  cartTotal,  } = useCart();  if (isEmpty) return <p className='empty'>Your cart is empty</p>;  return (  <>  <h1>Your Cart</h1>  <div className='cartTbl'>  <div className='cartUl'>  <div className='crtItem1'>Count</div><div className='crtItem1'> Name</div><div className='crtItem1'> Price</div><div className='crtItem1'></div>  </div>  {items.map((item) => (  <div className="cartUl">  <div key={item.id} className='crtItem1'>  {item.quantity}  </div>  <div className='crtItem1'>  {item.name}  </div>  <div className='crtItem1'>  {Math.ceil(((item.price \* (100 - item.discount)) / 100)\*100)/100}$  </div>  <div className='crtItem1'>  <button onClick={() => updateItemQuantity(item.id, item.quantity - 1)}className="crtbt">  -  </button>  <button onClick={() => updateItemQuantity(item.id, item.quantity + 1)} className="crtbt">  +  </button>  <button onClick={() => removeItem(item.id)}className="crtbt">&times;</button>  </div>  </div>  ))}  <p className='total'>Total ptice:{Math.ceil(cartTotal\*100)/100}</p>  <form>  <input type="text" className="buytxt" value="Name"/>  <input type="text" className="buytxt" value="Comments"/>  <input type="text" className="buytxt" value="Your email address..."/>  <button className='buy'>Buy</button>  </form>  </div>  </>  );  }  export const Crt=()=>{  return(  <html>  <header>  <div className="header">  <Link to="/" className="logo">Special Tea</Link>  <ul className="container">  <Link to='/Assort' className="Item1">Assortment</Link>  <Link to='/Cart' className="Item1">Your Cart</Link>  <Link to='/About' className="Item1">About Us</Link>  </ul>  </div>  </header>  <CartProvider>  <Cart/>  </CartProvider>  <footer>  <div className="fTop">  <ul className="footerTop">  <li>Shop</li>  <hr className="hr"/>  <Link to='/Assort' className='lnk'>All</Link>  <Link to='/Assort' className='lnk'>Tea</Link>  <Link to='/Assort' className='lnk'>Additives</Link>  </ul>  <ul className="footerTop">  <li>Share</li>  <hr className="hr"/>  <a href="https://www.instagram.com/" className='lnk'>Instagram</a>  <a href="https://google.com" className='lnk'>Site</a>  </ul>  <ul className="footerTop">  <li>Account</li>  <hr className="hr"/>  <Link to='/Cart' className='lnk'>Cart</Link>  </ul>  <ul className="footerTop">  <li>Legal</li>  <hr className="hr"/>  <li className='lnk'>Terms & Conditions</li>  <li className='lnk'>Privaty Policy</li>  </ul>  </div>  </footer>  </html>  );  }  export default Crt;  Search.js  import React from "react";  export class Search extends React.Component {  onChange = (e) => {  this.setState(() => this.props.search(e.target.value));  };  onClick = (target) => {  this.setState(() => this.props.searchParameter(target));  };  render() {  return (  <div>  <input className="search" type="search" placeholder="Search..." onChange={this.onChange}/>  </div>  );  }  } |