**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**Белорусский национальный технический университет**

**Факультет информационных технологий и робототехники**

**Кафедра «Программное обеспечение информационных систем и технологий»**

**ОТЧЕТ**

о прохождении преддипломной практики на ООО “ФордэКонсалтинг”

**Составил: Д.А.Андреев**

Студент группы 10701217 *(дата, подпись)*

(3 курс)

**Руководитель от предприятия: \_\_\_\_\_\_\_\_\_\_\_**

*(дата, подпись)*

**Руководитель от университета:** *(дата, подпись)* **Ю.В. Полозков**

**Минск 2021**

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 3](#_Toc533121602)  
[1. ИСПОЛЬЗУЕМЫЕ ТЕХНОЛОГИИ 5](#_Toc533121603)

[1.1 Front-end разработка. Основные характеристики 5](#_Toc533121604)

[1.1.1 React Js 6](#_Toc533121605)

[1.1.2 Webpack 8](#_Toc533121606)

[1.1. 3 Gulp. 10](#_Toc533121607)

[2. Обзор программно-техническиз средств 12](#_Toc533121609)

[2.1 React как инструмент для разработки приложений 12](#_Toc533121615)

[2.2 Библиотека React Virtualized 13](#_Toc533121616)

[2.3 Методы, используемые в библиотеке React Virtualized 16](#_Toc533121617)

[2.3.1 Методы для работы с таблицами 18](#_Toc533121618)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 20](#_Toc533121623)

## ВВЕДЕНИЕ

Никому не секрет, что интернет стал для нас второй реальностью, в которой мы проводим большее количество своего свободного времени. Каждый делает это по-разному, кто-то зависает в социальных сетях (Facebook, ВКонтакте, Одноклассники), кто-то заядлый блоггер и постоянно пишет целые статьи или же просто описывает события, которые с ним произошли в таких блогах как Live Journal, Twitter и другие. В интернете хватает сайтов на самые разные тематики, и предлагают человеку широкий круг возможностей от простого чтения сайтов до заработка денег, не выходя из комнаты своей квартиры. Также у любого предприятия по мере его развития со временем возникает потребность в контроле и отслеживании сетевой активности его узлов, серверов и прочих сетевых устройств. В мире программного обеспечения существует немало решений подобного рода задач и, как правило, некоторые из них активно используются предприятием. Однако часто системным администраторам необходим дополнительный специфический функционал, не реализуемый существующими приложениями.

Влияние глобальной компьютерной сети Internet на современный мир не имеет исторических аналогов. Его сегодняшний день - это начало эпохи электронного проникновения во все сферы человеческой жизни, это нечто большее, чем просто маркетинговая кампания, это основа новой философии и новой деловой стратегии.

Web-технология полностью перевернула представления о работе с информацией, да и с компьютером вообще. Оказалось, что традиционные параметры развития вычислительной техники - производительность, пропускная способность, емкость запоминающих устройств - не учитывали главного "узкого места" системы - интерфейса с человеком. Устаревший механизм взаимодействия человека с информационной системой сдерживал внедрение новых технологий и уменьшал выгоду от их применения. И только когда интерфейс между человеком и компьютером был упрощен до естественности восприятия обычным человеком, последовал беспрецедентный взрыв интереса к возможностям вычислительной техники.

Создание Web-сайтов является одной из важнейших технологий разработки ресурсов Internet. Хороший сайт, вбирая в себя всю полезную информацию, является лучшей визитной карточкой и коммерческой фирмы и образовательного учреждения, работая на них в любое время суток.

Так как же создаются такие сайты? Как организовать площадку для столь большого количества людей? Какие технологии нужно знать, что бы разработать web приложение? Все эти вопросы раскрыты в данной работе.

## 1. ИСПОЛЬЗУЕМЫЕ ТЕХНОЛОГИИ

# 1.1 Front-end разработка. Основные характеристики

Front-end разработка (разработка пользовательского интерфейса) - это практика создания HTML, CSS и JavaScript для веб-сайта или веб-приложения, чтобы пользователь мог напрямую видеть и взаимодействовать с ними. Задача, связанная с разработкой интерфейса, заключается в том, что инструменты и методы, используемые для его создания, постоянно меняются, и поэтому разработчик должен всегда быть в курсе, как эта сфера развивается.

Любой сайт представляет собой совокупность определенных слоёв - структур, данных, дизайна, контента и функциональности. Создание функциональности сайта, ориентированной на пользователя, - цель frontend разработки. Используя комбинацию языков разметки, дизайна и клиентских сценариев и фреймворков, frontend разработчик создает среду для всего, что пользователи видят и затрагивают: контент, макет и т.п. При найме стороннего разработчика полезно иметь широкое представление о том, как функционирует интерфейс сайта, что делает разработчик frontend, его обязанности и то, что его навыки могут внести свой вклад в ваш проект.

В последнее время front-end разработка стремительно развивается и сейчас ушла далеко вперёд. От банальной верстки html-страниц при помощи CSS c минимальной динамикой, задаваемой javascript/jquery, практически не осталось и следа. Если вы не работали хотя бы с одним из сборщиков front-end проектов ([grunt](http://gruntjs.com/" \t "_blank), [gulp](http://gulpjs.com/), [webpack](https://webpack.github.io/)), не слышали слова "транспайлер" ([babel](https://babeljs.io/" \t "_blank)), не знаете, что на front end тоже можно писать тесты ([jasmine](http://jasmine.github.io/" \t "_blank), [mocha](https://mochajs.org/), [chai](http://chaijs.com/)), и запускать их автоматически в любом браузере ([karma](https://karma-runner.github.io/" \t "_blank)), то вы весьма сильно отстали от жизни. А ведь есть ещё [React.js](https://facebook.github.io/react/" \t "_blank) (библиотека для построения интерфейсов), [ES6](http://www.ecma-international.org/ecma-262/6.0/" \t "_blank) (новейший стандарт языка javascript, который ещё не поддерживается большинством браузеров, но если очень хочется, то уже можно использовать в production-коде), [Node.js](https://nodejs.org/" \t "_blank) (платформа, позволяющая исполнить мечту любого front-end разработчика - писать не только front end, но и back end проекта на любимом javascript) и многое-многое другое.

Но как и любое новшество, данный скачок в развитии имеет две стороны медали. Мы попытаемся разобраться, какие есть недостатки и достоинства в работе с самыми новыми front-end технологиями и как постараться избежать проблем при старте работы с ними.

Работая над проектами, мы успели испробовать не только все вышеназванные, но и многие другие инструменты front-end разработчиков. Это был длинный и тернистый путь, и теперь мы хотим поделиться полученным опытом, дабы помочь вам приоткрыть дверь в мир современной javascript-разработки.

Для создания проекта на React и написанием на нем новых компонентов можно использовать сборщики Gulp или Webpack. Они позволяют поделить код на отдельные модули, упаковать их в нужные пакеты и оптимизировать время загрузки. При этом, Gulp – это менеджер, а Вебпак – бандлер (сборщик модулей). Рассмотрим каждый из них подробнее.

# 1.1.1 React Js

React Js – это JavaScript библиотека для построения пользовательских интерфейсов, разработанный программистами Facebook. Он предназначен для создания интерфейсов. Здесь важно учитывать, что React – это только система представления, в отличие от полноценного фреймворка, например, Angular.

React предоставляет разработчикам языковые шаблоны и функции обратной связи, позволяющие отрисовать HTML. Связки HTML-JS называются компонентами, которые запоминают свое внутреннее состояние. В React используется компонентный подход, что позволяет обходится без шаблонов, моделей, контроллеров и т.д.

Результат работы в React всегда будет зависеть от составляющих его компонентов. Особенностью этого фреймворка является JSX (сочетание Javascript и HTML). Что позволяет видеть происходящие изменения в одном месте. После обработки всех процессов в результате мы получаем чистый JS.

React может использоваться практически в любом проекте и подойдет как для крупного, так и малого бизнеса. Среди разработчиков существует немало споров по поводу использования этого продукта. Как и любой инструмент React имеет свои преимущества и недостатки.

Преимущества:

1. Легко представить конечный результат. Посмотрев на исходный код, можно представить, как будет отрисован нужный компонент. Зная состояние кода, можно с точностью определить результат отрисовки. Это важно при работе в команде, так как не нужно отслеживать весь процесс разработки проекта.
2. Простота использования. На React просто писать исходный код, и на нем же легко тестировать продукт.
3. Связка JavaScript и HTML в JSX позволяет быстрее и проще работать с компонентами. Следить за многоуровневым потоком JS-HTML-JS не доставляет большого удовольствия программистам. Объединение функциональной части с разметкой и их упаковка в отдельный компонент сделает код лучше.
4. Рендеринг на сервере. React можно визуализировать на сервере ещё до знакомства сайта с его потенциальными клиентами.
5. Создание интерфейса (UI) происходит на основе отдельных компонентов. Это будущее веб-разработки.

Недостатки:

1. Использование глобальных состояний. За счет компонентного подхода иногда глобальные состояния приходится пробрасывать сквозь 10 компонентов, а это трата времени на всего лишь одну константу.
2. Некоторые классические приложения не поддерживают React. Чтобы избежать этой ситуации применяют дополнительные плагины и скачивают расширения. Однако, из-за большого веса, в React их лучше использовать по минимуму.
3. React занимает достаточно места из-за большого веса по сравнению с подобным ему VueJS.

Почему стоит использовать React?

1. Прекрасно подходит для командной разработки, где осуществляется точное следование шаблону и соблюдение пользовательского интерфейса (UI).
2. UI код прост в сопровождении и легко читаем.
3. Разработка интерфейса происходит на основе отдельных компонентов – это прогрессивная тенденция в современном программировании.

# 1.1.2 Webpack

Приложения на JavaScript постоянно меняются и усложняются. Для наиболее удобной работы используется «сборщик» (бандлер) front-end проектов - Webpack. Он имеет открытый исходный код и позволяет решать большой спектр задач. С помощью этого инструмента программисты могут собрать требуемые ресурсы для проекта и провести упаковку и компиляцию необходимых элементов.

С помощью Webpack используются как файлы своей системы, так и иные библиотеки. Такая модульная система лучше организует проект, который разбит на отдельные, маленькие части.

Достоинства:

1. Благодаря Webpack в браузере используются различные модули http, events и т.д. Однако, если модуль все-таки несовместим с браузером, его можно заменить другим или оставить поле пустым – вариантов масса.
2. Этот инструмент имеет большое разнообразие плагинов для обработки кода. Позволяет сгенерировать сразу несколько файлов, в которых содержится определенное число плагинов.
3. С помощью Вебпака может выполнятся дополнительная обработка первоначальных файлов, благодаря сторонним загрузчикам и плагинам. - Webpack легко вызывается из любого диспетчера задач.
4. Возможность управлять и выполнять предварительную обработку данных проекта: файлы JS, таблицы стилей, изображений, шрифтов и т.д.
5. Инструмент позволяет продвинуто разделять код на составные части, загрузка которых осуществляется тогда, когда это необходимо браузеру.
6. Вебпак выбирают при работе с одностраничными приложениями.

Недостатки:

1. Сложен для новичков. Для тех, кто уже давно знаком с Webpack, разобраться достаточно просто.
2. Из-за того, что система постоянно меняется и дорабатывается, часть документации устаревает.
3. Специфика работы с CSS-файлами на начальном этапе кажется очень запутанной.

Гибкость настройки Webpack и его потенциал в программировании стали очень привлекательными для разработчиков. Он используется также и для управления кодом, предназначенного для работы на стороне заказчика. Многие клиентские библиотеки поддерживают различные модули и плагины, что приоткрывает новые перспективы в разработке.

Webpack – гибко настраиваемая система, которая постоянно совершенствуется и дорабатывается. Этот модульный упаковщик позволяет упаковывать модули во едино для контроля над файлами. На сегодняшний момент Вебпак – один из самых востребованных инструментов JS-разработки.

# 1.2. 3 Gulp.

Gulp – менеджер задач, который автоматизирует повторяющиеся процессы: сборка и оптимизация CSS- и JS-файлов, автоматическое обновление браузера, поднятие сервера и другие. Благодаря Gulp процессы разработки ускоряются и максимально оптимизируются. Инструмент Gulp создавался как ответвление от проекта Grunt, и он превзошел своего родителя.

Данный менеджер работает на Node.js платформе, исходный код прописывается на JavaScript. Что касается самого инструмента, возможностей у него нет так много. Однако, имеет большое разнообразие плагинов, которые можно между собой комбинировать в любой последовательности и количестве. Чтобы запустить нужную задачу, используется командная строка.

Одним из ощутимых достоинств Gulp служит использование потоков. Они передают данные поочередно от одной функции к другой, которые взаимодействуют определенным образом с этими данными. Такая система позволяет ускорить работу сборки.

Gulp поvогает решить следующие задачи:

1. Создание веб-сервера, а также обновление страниц в браузере при любых изменениях и сохранение проекта.
2. Внедрение в разработку препроцессоров (JS, CSS, HTML) для ускорения верстки.
3. Объединение, оптимизация и уменьшение объема исходного кода.
4. Автоматическое создание приставок к названию CSS-файлов для поддержки стилей браузеров.
5. Управление файлами и папками.
6. Выгрузка проекта на внешний сервер.
7. Копирование файлов проекта, создание резервных копий.
8. Создание проектов любой сложности и многое другое.

Проект, который использует Gulp имеет в корне файл gulpfile.js, содержащий подробные инструкции по управлению системой. Это достаточно большие файлы, но не стоит их пугаться, система сборки весьма простая, а все инструкции являются однотипными [4].

# 

# 2. ОБЗОР ПРОГРАММНО-ТЕХНИЧЕСКИХ СРЕДСТВ

# 2.1 React как инструмент для разработки приложений

Для разработки web-приложения использовался язык JavaScript с использованием библиотеки React для создания графического интерфейса. Одной из отличительных особенностей React является возможность использовать JSX, язык программирования с близким к HTML синтаксисом, который компилируется в JavaScript. Компонентно-ориентированный подход, возможность с легкостью изменять имеющиеся компоненты и переиспользовать код превращают React разработку в непрерывный процесс улучшения. Компоненты, которые были созданы во время работы над тем или иным проектом, не имеют дополнительных зависимостей. Таким образом, ничто не мешает использовать их снова и снова в проектах разного типа. Весь предыдущий опыт может быть с легкостью применен при работе над новым сайтом или даже при создании мобильного приложения. Используя передовые возможности, такие как Virtual DOM или изоморфный JavaScript, React разработчики могут с высокой скоростью создавать высокопроизводительные приложения, несмотря на уровень их сложности.

Возможность с легкостью заново использовать уже имеющийся код повышает скорость разработки, упрощает процесс тестирования, и, как результат, понижает затраты. Тот факт, что эта библиотека разрабатывается и поддерживается высококвалифицированными разработчиками набирает все большую популярность с каждым годом, дает основания надеяться, что тенденция к дальнейшим улучшениям продолжится.

Так же, согласно опросам популярность React выросла почти на 150% с 2017 до 2018 года. На самом деле ничего удивительного. Среди прочего, это стало возможным благодаря его растущей экосистеме, основанной на компонентах.

Как искать хорошие инструменты для разработки веб-проектов? Один из способов поиска заключается в анализе статистики проектов из интересующей вас сферы на GitHub. Если, например, некая библиотека пользуется популярностью, то это, по меньшей мере, говорит о том, что на неё стоит взглянуть. Вполне возможно, что одна из таких библиотек окажется именно тем, что вам нужно.

# 2.2 Библиотека React Virtualized

Библиотека React, представляет собой мощный JavaScript-фреймворк, упрощающий работу программистов, занимающихся веб-разработкой. Однако, если вы работали какое-то время с React JS, вы согласитесь с тем, что для того, чтобы в полной мере раскрыть возможности этого фреймворка, не помешает ещё несколько дополнительных библиотек.

Существует много различных библиотек. Рассмотрим некоторые из них:

1. Material UI - это библиотека, которая позволяет создавать приложения в стиле Google Material Design с использованием компонентов React. Она упрощает веб-разработку, создание привлекательных пользовательских интерфейсов и одностраничных приложений.

2. Ant-Design - это библиотека, основанная на React, с помощью которой в проект можно добавлять симпатичные элементы управления. Она разработана специально для создания настольных приложений, с ней очень приятно работать, а то, что получается в результате, позволяет говорить о том, что и пользователям приложений это понравится.

3. StoryBook позволяет создавать компоненты пользовательского интерфейса React-приложений и заниматься настройкой их внешнего вида в интерактивном графическом окружении, которое упрощает работу с компонентами React.

4. Enzyme - это легковесная библиотека для тестирования, построенная на базе React, которая служит в качестве JS-утилиты, позволяющей тестировать React-проекты. Кроме того, с её помощью можно задавать утверждения для проверки вывода компонентов React и осуществлять другие операции с интересующими разработчика данными.

Для реализации проекта используется одна из популярных библиотек - React Virtualized. Это близкий к совершенству набор компонентов React для виртуализации объемных наборов данных. Она предоставляет несколько повторно используемых компонентов для отображения списков, гридов и табличных данных. Вы также можете создать свой собственный экранирующий компонент, как это сделано в Twitter, если желаете что-то более специфическое для вашего конкретного случая. Проект набрал почти 10 тысяч звезд, на популярном Интернет-ресурсе, что говорит о его популярности и полезности.

На проекте используются компоненты таблиц и гридов. К сожалению, они идут стандартными и для многих проектов не подходят. Так как для реализации проекта не было достаточно этого стандартного функционала, грид пришлось переписать. Изначально, в видимой зоне грида некорректно отображались элементы: одни из них обрезались, другие при скроллинге зависали, а некоторые и вовсе не показывались.

# 3.3 Методы, используемые в библиотеке React Virtualized

Методы, используемые в библиотеке React Virtualized, предназначены для настройки корректного отображения данных. Перечислим основные из них:

* + 1. getOffsetForCell ({ alignment: ?string, columnIndex: ?number, rowIndex: ?number })

- возвращает смещения для данной ячейки.

* + 1. getTotalRowsHeight

- возвращает общую высоту строк.

* + 1. getTotalColumnsWidth

- возвращает общую ширину столбцов.

* + 1. handleScrollEvent ({ scrollLeft, scrollTop })

- этот метод обрабатывает событие прокрутки.

* + 1. recomputeGridSize ({ columnIndex: number, rowIndex: number })

- пересчитывает высоты строк и ширины столбцов после указанного индекса (по умолчанию 0). Эта функция должна вызываться, если размеры динамических столбцов или строк изменились, но больше ничего не изменилось. Поскольку Grid только получает columnCount и rowCount не имеет возможности обнаруживать, когда базовые данные изменены. Этот метод будет также заставить рендер цикла (через forceUpdate), чтобы гарантировать, что обновленный измерений отражена в представленном сетки.

* + 1. scrollToCell ({ columnIndex: число, rowIndex: число })

- прокрутка к ячейке

* + 1. scrollToPosition ({ scrollLeft, scrollTop })

- прокрутка к позиции. Полезно для анимации изменений позиции.

Свойство cellRenderer - отвечает за отрисовку одной ячейки, учитывая ее строку и индекс столбца.

Расширенное свойство cellRangeRenderer. Это полезно для ситуаций, где нужно написать свой визуальный cellRange.

Ниже приведен пример простого Grid. Сетка отображает массив объектов с фиксированными размерами строк и столбцов:

import React from 'react';

import ReactDOM from 'react-dom';

import { Grid } from 'react-virtualized';

// Grid data as an array of arrays

const list = […]

function cellRenderer ({ columnIndex, key, rowIndex, style }) {

return (

<div

key={key}

style={style}

>

{list[rowIndex][columnIndex]}

</div>

)

}

// Render your grid

ReactDOM.render(

<Grid

cellRenderer={cellRenderer}

columnCount={list[0].length}

columnWidth={100}

height={300}

rowCount={list.length}

rowHeight={30}

width={300}

/>,

document.метода getElementById('пример')

);

# 3.3.1 Методы для работы с таблицами

1. forceUpdateGrid

- принудительно перерисовывает внутренний Grid компонент. Используйте этот метод, если нужно вручную вызвать повторную отрисовку. Это может быть уместно, если базовые данные строки изменились, но сами размеры строк не изменились;

1. getOffsetForRow

- возвращает смещение для данной строки и выравнивания;

1. measureAllRows

- предварительное измерение всех строк таблицы. Этот метод гарантирует, что следующий вызов getTotalSize() возвращает точный размер (в отличие от только оцененного);

1. recomputeRowHeights (индекс: число)

- пересчитывает высоты строк и смещения после указанного индекса (по умолчанию 0). Этот метод также заставит перерисоваться (через forceUpdate), чтобы гарантировать, что обновленные измерения отражены в представленной таблице.

1. scrollToPosition (scrollTop: число)

- прокрутка до указанного смещения. Полезно для анимации изменений позиции.

1. scrollToRow (индекс: число)

- этот метод можно использовать для безопасной прокрутки назад к ячейке.

Компонент Table поддерживает следующие статические имена классов:

Таблица 3.1 – Статические имена классов

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| Таблица ReactVirtualized\_\_ | Главный (внешний) элемент |
| ReactVirtualized\_\_Table\_\_headerColumn | Заголовок ячейки (аналогично thead > tr > th) |
| ReactVirtualized\_\_Table\_\_headerRow | Строку заголовка (аналогично thead > tr) |
| Строка ReactVirtualized\_\_ Table\_\_ | Строки таблицы (сродни tbody > tr) |
| ReactVirtualized\_\_Table\_\_ rowColumn | Таблица столбец (сродни tbody > tr > td) |
| ReactVirtualized\_\_Table\_\_\_sortableHeaderColumn | Применяется к столбцам заголовка, которые можно сортировать |
| ReactVirtualized\_\_Table\_\_sortableHeaderIcon | Индикатор сортировки SVG |

headerRowRenderer - это расширенное свойство. Оно полезно для ситуаций, когда требуются дополнительные крючки на таблице, оказание дополнительных настраиваемых элементов пользовательского интерфейса. Если начать с копирования функции [defaultTableHeaderRowRenderer](https://z5h64q92x9.net/proxy_u/en-ru.ru/https/github.com/bvaughn/react-virtualized/blob/master/source/Table/defaultHeaderRowRenderer.js) , эта функция принимает следующие именованные параметры:

Таблица 3.2 –Параметры функции defaultTableHeaderRowRenderer

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| имя класса | Имя класса заголовка |
| столбцы | Массив узлов React |
| стиль | Объект стиля заголовка |

rowRenderer - это расширенное свойство,которое полезно для ситуаций, когда требуются дополнительные крепления Table (например, интеграция с библиотекой как react-sortable-hoc). Если требуется больше настроек, можно скопировать функцию  [defaultTableRowRenderer](https://z5h64q92x9.net/proxy_u/en-ru.ru/https/github.com/bvaughn/react-virtualized/blob/master/source/Table/defaultRowRenderer.js). Эта функция принимает следующие именованные параметры:

Таблица 3.3 – Параметры функции  [defaultTableRowRenderer](https://z5h64q92x9.net/proxy_u/en-ru.ru/https/github.com/bvaughn/react-virtualized/blob/master/source/Table/defaultRowRenderer.js).

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| имя класса | Имя класса уровня строки |
| столбцы | Массив узлов React |
| Указатель | Индекс строки |
| isScrolling | Логический флаг, указывающий, если Table в настоящее время прокручивается |
| onRowClick | Дополнительный ряд onClick обработчик |
| onRowDoubleClick | Дополнительный ряд onDoubleClick обработчик |
| onRowMouseOver | Дополнительный ряд onMouseOver обработчик |
| onRowMouseOut | Дополнительный ряд onMouseOut обработчик |
| rowData | Данные строк |
| Стиль | Объект стиля уровня строки |

Ниже приведень простой Table пример. Эта Таблица содержит только 2 столбца, каждый из которых содержит простую строку. Оба имеют фиксированную ширину и ни один из них не сортируется.:

import React from 'react';

import ReactDOM from 'react-dom';

import { Column, Table } from 'react-virtualized';

import 'react-virtualized/styles.css'; // only needs to be imported once

// Table data as an array of objects

const list = [

{ name, description }

// And so on...

];

// Render your table

ReactDOM.render(

<Table

width={300}

height={300}

headerHeight={20}

rowHeight={30}

rowCount={list.length}

rowGetter={({ index }) => list[index]}

>

<Column

label='Name'

dataKey='name'

width={100}

/>

<Column

width={200}

label='Description'

dataKey='description'

/>

</Table>,

document.getElementById('example')

);

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

* 1. Бейтс, Марк CoffeeScript. Второе дыхание JavaScript / Марк Бейтс. - М.: ДМК Пресс, 2016. - 310 c.
  2. Бенедетти, Р. Фрейн – HTML5 CSS3 – разработка сайтов для любых браузеров и устройств/ Р. Бенедетти. - М.: Питер, 2012. - 737 c.
  3. Вальтер, Штефен Создание приложений для Windows 8 с использованием HTML5 и JavaScript / Штефен Вальтер. - М.: ДМК Пресс, 2013. - 344 c.
  4. JS web frameworks benchmark. [Электронный ресурс]. – Режим доступа: <https://habr.com/company/netologyru/blog/327294/>
  5. Frontend-developer [Электронный ресурс]. – Режим доступа: <https://itvdn.com/ru/specialities/frontend-developer>
  6. Кантелон, М. Node.js в действии / М. Кантелон. - М.: Питер, 2015.
  7. Козловский, Павел Разработка веб-приложений с использованием AngularJS / Павел Козловский , Питер Бэкон Дарвин, Павел Козловский. - Москва: СИНТЕГ, 2014. - 394 c.
  8. Крокфорд, Д. JavaScript. Сильные стороны / Д. Крокфорд. - М.: Питер, 2013. - 117 c.
  9. Макфарланд, Дэвид JavaScript и jQuery. Исчерпывающее руководство (+ DVD-ROM) / Дэвид Макфарланд. - М.: Эксмо, 2012. - 688 c.

1. Сухов, Кирилл Node.js. Путеводитель по технологии / Кирилл Сухов. - М.: ДМК Пресс, 2015. - 416 c.
2. Флэнаган, Дэвид JavaScript. Подробное руководство / Дэвид Флэнаган. - М.: Символ-плюс, 2013. - 714 c.
3. Фримен, Адам jQuery для профессионалов / Адам Фримен. - М.: Вильямс, 2013. - 960 c.
4. Хэррон, Дэвид Node.js Разработка серверных веб-приложений на JavaScript / Дэвид Хэррон. - М.: ДМК Пресс, 2012. - 987