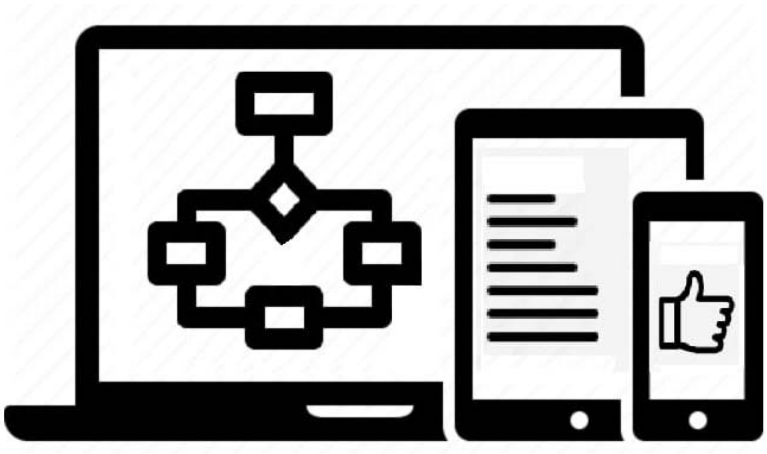
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Программное обеспечение информационных технологий»

ТЕХНОЛОГИИ ИНТЕРНЕТ-ПРОГРАММИРОВАНИЯ

Методические рекомендации к лабораторным работам для студентов направлений подготовки 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» очной формы обучения



Могилев 2024

УДК 004.4(07)
ББК 32.973я73
Т384

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Программное обеспечение информационных технологий» «26» апреля 2024 г., протокол № 10

Составитель канд. техн. наук Ю. В. Вайнилович

Рецензент канд. техн. наук, доц. А. П. Прудников

Методические рекомендации содержат требования к лабораторным работам по дисциплине «Технологии интернет-программирования» для студентов направлений подготовки 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» очной формы обучения.

Учебное издание

ТЕХНОЛОГИИ ИНТЕРНЕТ-ПРОГРАММИРОВАНИЯ

Ответственный за выпуск	В. В. Кутузов
Корректор	А. Т. Червинская
Компьютерная верстка	М. М. Дударева

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 16 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

Введение

Цель учебной дисциплины состоит в формировании у студентов глубоких теоретических знаний и практических навыков в области web-программирования, глубоком представлении об основных технологиях и инструментах, используемых при разработке веб-сайтов и приложений как на стороне клиента, так и на стороне сервера.

Методические рекомендации по дисциплине «Технологии интернет-программирования» к лабораторным работам предназначены для оказания помощи студентам при выполнении лабораторных работ по данной дисциплине. Они содержат задания для самостоятельного выполнения, инструкции, пояснения и рекомендации, которые помогают студентам освоить конкретные технологии и принципы программирования, связанные с разработкой интернет-приложений.

По результатам выполнения каждой лабораторной работы студент предоставляет отчет, который содержит:

- титульный лист;
- цель работы;
- постановку задачи;
- описание результатов выполненной работы.

1 Лабораторная работа № 1. Фиксированная верстка готового дизайн-макета

Цель работы: освоить работу в Figma, освоить технологии HTML, CSS.

Теоретический материал

Изучить основы работы в Figma с использованием [1–4].
Ознакомиться со следующим материалом.
1 Основы CSS:
- https://webref.ru/course/css-basics.

2 Основы позиционирования элементов:
- https://webref.ru/course/position;
- https://webref.ru/course/block-model;
- https://webref.ru/course/block-inline.

3 CSS grid layout:
- основные понятия Grid Layout;
- полное руководство по CSS Grid;
- верстка на Grid в CSS;
- https://css-tricks.com/snippets/css/complete-guide-grid;
- https://css-tricks.com/snippets/css/a-guide-to-flexbox/.

4 Flexbox:
- https://webref.ru/layout/flexbox-tutorial;
- https://habr.com/ru/post/467049/;
- CSS: Flexbox;
- шпаргалка по Flexbox.

Задание для самостоятельного выполнения.
Разработать дизайн-макет браузерной версии лендинга. Сверстать спроектированный лендинг. Варианты тем лендингов выбираются из приложения А.

Требования к верстке

Верстка валидная.

Для проверки валидности верстки используйте сервис https://validator.w3. org/. Валидной верстке соответствует надпись «Document checking completed. No errors or warnings to show.».

Верстка семантическая.

Запрещается использование CSS-фреймворков (bootstrap, foundation и т. д.). **Запрещается** использование JS-фреймворков (Angular, React, Vue и т. д.). **Запрещается** использование устаревших библиотек (jQuery и т. д.).

Для создания многоколоночных структур, или элементов, имеющих относительное горизонтальное расположение, должно быть использовано одно из свойств:

– display: flex;
– display: grid;
– display: inline-block.

Основные блоки должны быть точно расположены на заданной ширине экрана так, как в макете Figma.

Изображения, логотипы (если они есть) должны быть расположены в рамках логического контейнера с правильным подходом по центрированию и расположению. Допускается незначительное отклонение от макета в угоду сеточной или колоночной структуре.

Иконки, картинки должны сохранять идеальное расстояние до начала соответствующего им текста.

Иконки, картинки должны сохранять свои пропорции.

Если использован правильный шрифт, следует проверить высоту текста – он должен соответствовать исходнику. Ширина может варьироваться. Но общепринятой практикой является добавление свойства межбуквенного интервала (letter-spacing) тексту заголовков, девиза (motto) или цитат.

Если в строке несколько объектов визуально одинаковой ширины, то ширина содержащих их блоков должна быть одинаковой. Разница размеров изображений не имеет значения, важно совпадение размеров блоков. Если в макете ширина блоков разная, то делать ее все равно нужно одинаковой.

Некоторые элементы должны быть интерактивными. «Интерактивный» означает, что у кнопки или элемента появляется визуальный эффект или анимация (на ваше усмотрение и исходя из макета: анимация курсора, изменение цвета заднего фона, затемнение, нижнее подчеркивание, изменение шрифта) при каких-либо действиях пользователя, например, при наведении курсора. Использовать JavaScript для обработки пользовательских событий в данном задании не обязательно. Обычно такой эффект реализуют при помощи псевдокласса :hover и следующих свойств:

– cursor: pointer;
– background;
– text-decoration: underline;
– color.

2 Лабораторная работа № 2. Адаптивная верстка готового дизайн-макета. Технологии CSS Grid и Flexbox

Цель работы: овладеть техниками и методологиями адаптивного дизайна, выработать умение применять их для создания гибких и отзывчивых веб-интерфейсов.

Теоретический материал

CSS3-медиазапросы (https://html5book.ru/css3-mediazaprosy/).

CSS3 Адаптивный веб дизайн. – Медиазапросы (https:// www.schoolsw3.com/css/css_rwd_mediaqueries.php).

Задание для самостоятельного выполнения.

Разработать дизайн-макеты для планшетной и мобильной версий лендинга. Сверстать спроектированные страницы.

Требования к верстке

Должны быть выполнены все требования лабораторной работы № 5 до порогового значения (меньше 320px). Это означает, что отступы, размеры блоков, и прочее не должны уходить за правый край экрана и не должен появляться горизонтальный скролл.

Задание оценивается путем изменения размеров окна браузера Google Chrome или подключением эмуляции устройств через панель разработчика (DevTools -> Toogle Device Toolbar), выбрав значение ширины экрана.

При проверке должна отсутствовать вертикальная полоса прокрутки, т. к. она «съедает» часть пространства отзывчивой верстки своей шириной. Чтобы ее отключить, необходимо выбрать режим эмуляции Responsive, а также установить тип устройства Mobile (рисунок 1). Если тип устройства не отображается, в верхней панели device toolbar нажмите на три точки справа и выберите Add device type.

Рисунок 1 – Режим эмуляции

«Responsive» – это размеры, заданные в относительных величинах от ширины окна или родительского блока, которые плавно меняют свои значения при уменьшении или увеличении окна браузера. Главное, чтобы при наложении картинки, например, в 768px на макет шириной 768px, размеры или отступы совпадали.

3 Лабораторная работа № 3. Основы программирования на JavaScript

Цель работы: разработать дизайн экранов, выбрать подходящие элементы управления, определить навигацию и создать прототипы интерфейса для мобильного приложения.

Теоретический материал

Учебник по javascript (https://learn.javascript.ru/array-methods).

15 методов работы с массивами в JavaScript, которые необходимо знать в 2020 году (https://habr.com/ru/companies/plarium/articles/483958/).

Документация по синтаксису javascript (https://doka-guide.vercel.app/js/).

Справочник по javascript. Встроенные объекты. Array (https:// developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/ Array).

Задание для самостоятельного выполнения.

Задача 1. Реализовать функцию check(str, bracketsConfig), которая для заданной последовательности скобок вернет true, если она правильная, и false – в противном случае.

Второй параметр bracketsConfig – массив пар открытых-закрытых скобок. Каждый подмассив включает в себя всего два элемента – открывающую и закрывающую скобки.

```
check('()', [['(', ')']]) // -> true
check('((()))()', [['(', ')']]) // -> true
check('())(', [['(', ')']]) // -> false
check('([{}])', [['(', ')'], ['[', ']'], ['{', '}']]) // -> true
check('[(])', [['(', ')'], ['[', ']']]) // -> false
check('[]()', [['(', ')'], ['[', ']']]) // -> true
check('[]()(', [['(', ')'], ['[', ']']]) // -> false
// особый случай: открывающая и закрывающая скобка могут быть одинаковыми
check('||', [['|', '|']]) // -> true
check('|()|', [['(', ')'], ['|', '|']]) // -> true
check('|(|)', [['(', ')'], ['|', '|']]) // -> false
check('|()|(||)||', [['(', ')'], ['|', '|']]) // -> true
```

Задача 2. Ваша задача – реализовать так называемый алгоритм «сортировки полотенца».

Функция TowelSort должна ожидать матрицу любой формы, например:

```
[
[ 1, 2, 3 ],
[ 4, 5, 6 ],
[ 7, 8, 9 ],
]
```

Следующая матрица должна быть «отсортирована» по:

[1, 2, 3, 6, 5, 4, 7, 8, 9]

Задача 3. Ваша задача – написать функцию, которая декодирует азбуку Морзе и возвращает строку.

Длина строки ввода кратна 10.

Каждая буква алфавита закодирована точками (.) и тире (–). 10 означает точку (.), 11 означает тире (–).

Длина каждой закодированной буквы равна 10.

Если длина закодированной буквы меньше 10, она дополняется 0.

Пробел в строке обозначается «**********».

Вывод: строка (декодированная).

Пример: me -> m === -- === 0000001111, e === . === 0000000010 -> 00000011110000000010

4 Лабораторная работа № 4. Основы программирования на TypeScript

Цель работы: приобрести знания и навыки, необходимые для разработки масштабируемых и поддерживаемых веб-приложений с использованием TypeScript.

Теоретический материал

Перед выполнением практического задания ознакомиться с [7].

Задание для самостоятельного выполнения.

Сделать копии приложений, разработанных в лабораторной работе № 3.

Добавить TypeScript в проект.

Настроить ESLint для работы с TypeScript.

Настроить Webpack для работы с Typescript.

Смигрировать приложения с JavaScript на TypeScript. Обязательно использовать:

- enum;
- interface;
- type;
- generics;
- union;
- private, public;
- partial, pick, readonly;
- тип any использовать запрещается.

5 Лабораторная работа № 5. Добавление интерактивности на web-страницу

Цель работы: добавить интерактивность ранее сверстанной web-странице с использованием JavaScript для реализации указанного в задании функционала.

Теоретический материал

Учебник по javascript (https://learn.javascript.ru/).

Задание для самостоятельного выполнения.

Добавить для лендинга, сверстанного в лабораторной № 2 следующую интерактивность:

- перевод страницы на два языка;
- подсветка активной кнопки (например en/ru);
- возможность переключения «светлой» и «темной» тем;

– сохранить выбранные пользователем настройки в LocalStorage (язык отображения страницы и светлая или темная тема);
– смена изображения на странице;
– добавление звука;
– новые элементы должны быть стилизованы.

6 Лабораторная работа № 6. Асинхронный JavaScript

Цель работы: научиться получать данные от API и отображать их на web-странице.

Теоретический материал

Using the Fetch API. (https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch).

Практическое ES6 руководство о том, как сделать HTTP запрос с помощью Fetch API. (https://jem-space.ru/praktichieskoie-es6-rukovodstvo-o-tom-kak-sdielat-http-zapros-s-pomoshchiu-fetch-api/).

Working with JSON. (https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON).

Что такое JSON.(https://habr.com/ru/post/554274/).

Задание для самостоятельного выполнения.

Разработать приложение, отображающее полученные от API фото. Добавить приложению поиск. При вводе поискового запроса изменяются фото, которые отображаются в приложении.

Варианты API.

1 Unsplash API:
– сайт: https://unsplash.com/developers;
– документация: https://unsplash.com/documentation.

2 Flickr API:
– сайт: https://www.flickr.com/services/;
– документация: https://www.flickr.com/services/api/flickr.photos.search. html.

Требования к верстке

Верстка адаптивная. Приложение хорошо выглядит при ширине страницы от 1920px до 768px.

Интерактивность элементов, с которыми пользователи могут взаимодействовать, изменение внешнего вида самого элемента и состояния курсора при наведении, использование разных стилей для активного и неактивного состояния элемента, плавные анимации.

В футере приложения есть ссылка на GitHub автора приложения, год создания приложения.

Представленные варианты страниц не для копирования, а как пример того, что можно сделать.

Технические требования

Работа приложения проверяется в браузере Google Chrome последней версии.

Можно использовать bootstrap, material design, css-фреймворки, html и css препроцессоры.

Не разрешается использовать jQuery, другие js-библиотеки и фреймворки.

Js-код приложения должен быть читаемым.

7 Лабораторная работа № 7. DOM Api

Цель работы: приобрести практические навыки навигации и управления DOM, понимание и применение CSS через JavaScript, а также обработки сложного поведения веб-документов.

Теоретический материал

Что такое Объектная Модель Документа (DOM)?

(https://developer.mozilla.org/ru/docs/Web/API/Document_Object_Model/Introduction).

Управление документами. (https://developer.mozilla.org/ru/docs/ Learn/ JavaScript/Client-side_web_APIs/Manipulating_documents).

Урок 7. JavaScript в веб-разработке (https://smartiqa.ru/courses/web/lesson-7-js).

Задание для самостоятельного выполнения.

Разработать виртуальную клавиатуру. Пример представлен на рисунке 2.

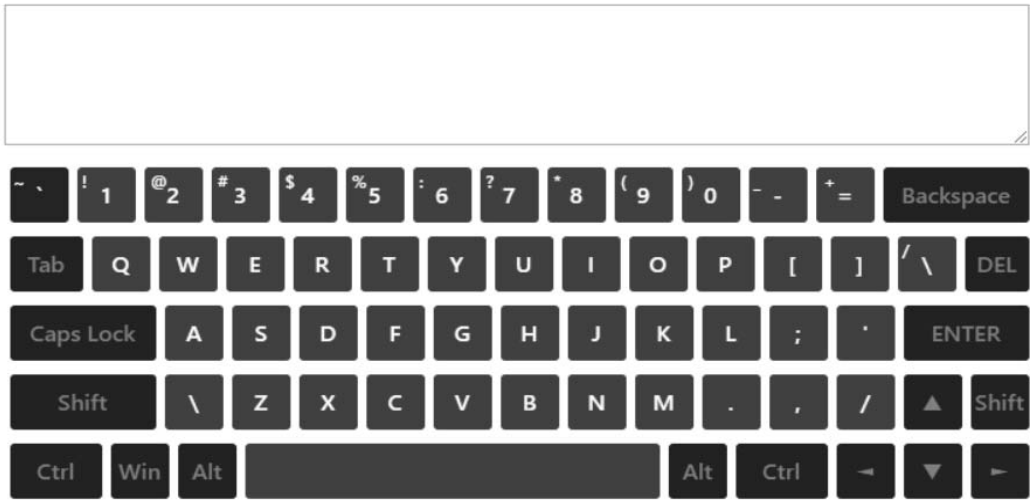


Рисунок 2 – Пример виртуальной клавиатуры

Требования к приложению

Дизайн – на ваше усмотрение.

Index.html должен быть пустым. Все необходимые элементы генерируются с использованием JS.

Если нажато несколько кнопок, то на виртуальной клавиатуре подсвечиваются все нажатые кнопки (также нет исключений для Ctrl, Alt и Shift).

Виртуальная клавиатура способна переключаться между двумя языковыми раскладками (английский + любой другой язык).

Назначение сочетания клавиш для переключения раскладки клавиатуры зависит от вас.

Кнопки на виртуальной клавиатуре отображают символы выбранного языка.

Приложение сохраняет выбранный язык после перезагрузки страницы и отображает клавиатуру на этом языке.

На странице должна быть указана комбинация клавиш для смены языка, чтобы пользователю было понятно, как переключать раскладку клавиатуры.

Нажатия клавиш анимированы.

Щелчки мышью по кнопкам на виртуальной клавиатуре и нажатия клавиш на физической клавиатуре должны вводить символы в текстовую область, расположенную на странице над виртуальной клавиатурой.

Нажатие клавиши со стрелкой вверх, вниз, влево или вправо вводит символ стрелки в поле ввода или реализует навигацию по текстовой области.

Нажатие Enter должно переместить текстовый курсор на следующую строку.

Клавиша Tab создает горизонтальный отступ.

Нажатие остальных функциональных клавиш на клавиатуре не приводит к вводу символов.

Клавиша Backspace удаляет символ перед текстовым курсором.

Клавиша Del удаляет символ после текстового курсора.

Клавиши Shift, Alt, Ctrl, Caps Lock и Space должны работать как на настоящей клавиатуре.

Технические требования

Работа приложения проверяется в браузере Google Chrome последней версии.

Запрещается использовать Angular/React/Vue и другие фреймворки.

Не разрешается использовать jQuery, другие js-библиотеки.

Js-код приложения должен быть читаемым.

8 Лабораторная работа № 8. Разработка SPA-приложения с использованием библиотеки React

Цель работы: научиться разрабатывать web-приложения с использованием библиотеки React.

Теоретический материал

Официальная документация по React (https://ru.reactjs.org/docs/getting-started.html).

Учебный курс по React (https://habr.com/ru/company/ruvds/blog/436032/);

Основной учебник React (https://code.mu/ru/javascript/framework/react/book/prime/).

React – пробрасываем данные из парент компонента в чайлд с помощью props (https://monsterlessons.com/project/lessons/react-probrasyvaem-dannye-iz-parent-komponenta-v-chajld-s-pomoshyu-props).

Полное руководство по useEffect (https://habr.com/ru/company/ruvds/blog/445276/).

Задание для самостоятельного выполнения.

У вашей бабушки скопилось большое количество елочных игрушек. Она попросила вас помочь их разобрать, чтобы выбрать те, которыми будет наряжать елку в этом году.

Зная, как бережно бабушка относится к этим игрушкам, вы отнеслись к ее просьбе очень внимательно.

Вы составили опись всех имеющихся игрушек, указав для каждой ее название, количество экземпляров, год покупки, форму (шар, фигурка, снежинка и т. д.), цвет, размер, отдельно отметили любимые бабушкины игрушки.

Теперь вам предстоит создать приложение, которое позволит отсортировать игрушки по названию и количеству экземпляров, найти игрушку по названию, сгруппировать игрушки по видам, добавлять игрушки в избранное и удалять из него, а также с интерактивной страницей, на которой выбранными игрушками можно украсить новогоднюю елку.

Функционал приложения

1 Страница с игрушками содержит карточки всех игрушек а также фильтры, строку поиска, поле для сортировки.

2 Карточка игрушки содержит ее изображение, название, текстом или условным значком обозначено количество экземпляров, год покупки, форма, цвет, размер, является ли игрушка любимой. Карточки игрушек добавляются динамически средствами JavaScript.

3 Добавление игрушек в избранное:

– кликая по карточке с игрушкой или по кнопке на ней, игрушку можно добавлять в избранное или удалять из избранного. Карточки добавленных в избранное игрушек внешне отличаются от остальных;

– на странице отображается количество добавленных в избранное игрушек. При попытке добавить в избранное больше 20 игрушек выводится всплывающее уведомление с текстом «Извините, все слоты заполнены».

4 Сортировка. Сортируются только те игрушки, которые в данный момент отображаются на странице:

– сортировка игрушек по названию в возрастающем и спадающем порядке;

– сортировка игрушек по году их приобретения в возрастающем и спадающем порядке.

5 Фильтры в указанном диапазоне «от» и «до»:

– фильтры по количеству экземпляров;

– фильтры по году покупки;

– для фильтрации в указанном диапазоне используется range slider с двумя ползунками. При перемещении ползунков отображается их текущее значение, разный цвет слайдера до и после ползунка. Range slider можно создать на основе input[type=range] или использовать для этой цели библиотеку noUiSlider или другую на ваш выбор.

6 Фильтры по значению. Выбранные фильтры выделяются стилем:

– фильтры по форме;

– фильтры по цвету;

– фильтры по размеру;

– можно отобразить только любимые игрушки;

– можно отфильтровать игрушки по нескольким фильтрам одного типа;

– для нескольких фильтров одного типа отображаются игрушки, которые соответствуют хоть одному выбранному фильтру. Например, можно отобразить снежинки и колокольчики; или белые, синие и красные игрушки; или большие и средние.

7 Можно отфильтровать игрушки по нескольким фильтрам разного типа. Для нескольких фильтров разного типа отображаются только те игрушки, которые соответствуют всем выбранным фильтрам. Например, можно отобразить только синие шары; или любимые белые и красные игрушки купленные в 1940–1960 гг.

Если игрушек, соответствующих всем выбранным фильтрам, нет, на странице выводится уведомление в человекочитаемом формате, например «Извините, совпадений не обнаружено».

8 Сброс фильтров:

– есть кнопка reset для сброса фильтров;

– кнопка reset сбрасывает только фильтры, не влияя на порядок сортировки или игрушки, добавленные в избранное. После использования кнопки reset фильтры остаются работоспособными;

– при сбросе фильтров кнопкой reset ползунки range slider сдвигаются к краям, значения ползунков возвращаются к первоначальным, range slider закрашивается одним цветом.

9 Сохранение настроек в local storage. Выбранные пользователем фильтры, порядок сортировки, добавленные в избранное игрушки сохраняются при перезагрузке страницы. Есть кнопка сброса настроек, которая очищает local storage.

10 Поиск:

– при открытии приложения курсор находится в поле поиска;

– автозаполнение поля поиска отключено (нет выпадающего списка с предыдущими запросами);

– есть placeholder;

– в поле поиска есть крестик, позволяющий очистить поле поиска;

– если нет совпадения последовательности букв в поисковом запросе с названием игрушки, выводится уведомление в человекочитаемом формате, например «Извините, совпадений не обнаружено»;

– при вводе поискового запроса на странице остаются только те игрушки, в которых есть указанные в поиске буквы в указанном порядке. При этом не обязательно, чтобы буквы были в начале слова. Регистр символов при поиске не учитывается;

– поиск ведется только среди игрушек, которые в данный момент отображаются на странице;

– если очистить поле поиска, на странице отображаются игрушки, соответствующие всем выбранным фильтрам и настройкам сортировки.

11 Дополнительный функционал на выбор:

– в процессе сортировки, фильтрации, поиска карточки с изображениями игрушек плавно меняют свое положение. Код для плавного перемещения карточек можно написать самостоятельно или использовать для этой цели какую-либо библиотеку. Библиотека может использоваться только для перемещения карточек. Фильтрацию, сортировку и поиск необходимо написать полностью самостоятельно;

– очень высокое качество оформления приложения и дополнительный, не указанный в пп. 1–10, сложный в реализации функционал, улучшающий качество приложения, удобство пользования им.

9 Лабораторная работа № 9. Разработка React- приложений на базе библиотек готовых компонентов

Цель работы: научиться разрабатывать web-приложения с использованием библиотек MaterialUI, bootstrap.

Теоретический материал

Официальная документация по MaterialUI (https://mui.com/material-ui/getting-started/).

Официальная документация по фреймворку Bootstrap (https://getbootstrap.com/).

Задание для самостоятельного выполнения.

Разработать приложение-викторину по определенной тематике.

Тематика викторины выбирается студентом самостоятельно и согласовывается с преподавателем.

Требования к приложению:

– на основании предложенных исходных данных средствами JavaScript генерируются вопросы двух типов и варианты ответов к ним;

– при этом вопросы идут последовательно, как они записаны в коллекции исходных данных, а варианты ответов создаются рандомно;

– всего на основании предложенных исходных данных необходимо сгенерировать 240 вопросов – по 120 вопросов каждого типа;

– вопросы викторины разбиты на группы (категории). В каждой категории десять вопросов;

– у вопросов одной категорий нет единой тематики или другого объединяющего их признака. Вопросы делятся на категории по порядку размещения данных в файле с исходными данными. Названия категорий – произвольные;

– при желании можно создать тематические категории;

– прохождение всех вопросов одной категории составляет один раунд игры. Данные про сыгранные раунды и их результаты, а также про настройки приложения хранятся в local storage;

– для каждого вопроса генерируется четыре варианта ответов. Пользователь выбирает ответ, кликая по карточке с ним;

– ответы представлены в виде карточек с некоторой текстовой или графической информацией;

– после выбора пользователем ответа, появляется индикатор, разный для правильных и неправильных ответов, выводится правильный ответ, появляется возможность перейти к следующему вопросу;

– после окончания раунда выводится его результат – количество вопросов, на которые был дан правильный ответ;

– результаты всех пройденных раундов отображаются на карточках категорий;

– для каждой сыгранной категории можно открыть страницу с результатами, на которой отображаются все картины категории – цветные или черно-белые в зависимости от того, были ли они угаданы правильно;

– сыгранный раунд можно пройти повторно, при этом вопросы будут повторяться, а варианты ответов будут другими;

– дизайн, оформление и функционал викторины – на усмотрение студента.

10 Лабораторная работа № 10. Архитектурный паттерн Redux

Цель работы: изучить и научиться применять архитектурный паттерн Redux в разработке приложений, освоение принципов однонаправленного потока данных, централизации состояния приложения и управления изменениями стейта с помощью Redux.

Теоретический материал

Введение в Redux & React-redux (https://habr.com/ru/articles/498860/).
Redux Toolkit (https://rajdee.gitbook.io/redux-toolkit-in-russian).
Справочник React. Redux Toolkit (https://reactdev.ru/libs/redux-toolkit/).

Задание для самостоятельного выполнения.

Задание выполняется на основе приложения, разработанного в лабораторной работе № 9. Реализовать управление состоянием приложения с помощью Redux Toolkit.

11 Лабораторная работа № 11. Архитектура веб-проектов на Node.js

Цель работы: приобрести знания и навыки, необходимые для разработки масштабируемых и поддерживаемых веб-приложений с использованием TypeScript.

Теоретический материал

Работа с файлами в NodeJS (https://metanit.com/web/nodejs/2.8.php).
Пакет nodemon (https://www.npmjs.com/package/nodemon).
Express 4.x – API Reference (https://expressjs.com/en/api.html#express).

Задание для самостоятельного выполнения.

Разработать серверную компоненту web-приложения, удовлетворяющую следующим требованиям.

На бэкенде должны выполняться следующие запросы:

– GET-сервис, который возвращает какие-то данные в формате JSON;
– POST-сервис, который возвращает какие-то данные в формате JSON;
– POST-сервис, который принимает какие-то данные;
– GET-сервис, который возвращает саму веб-страницу с фронтовым кодом;
– какой-либо сервис для получения данных в формате XML/HTML/JSON в зависимости от заголовка запроса Accept;
– исходные данные хранятся в JSON-файлах;
– при изменении JSON-файлов приложение должно продолжать работу.

На фронтенде должны выполняться следующие требования:

– получить с бэкенда данные по GET- и POST-сервисам;
– отобразить данные в любом виде (кнопки, списки, радиокнопки и т. д.);
– при каком-либо событии (нажатии на кнопку, выборе из списка и т. д.), в результате которого данные должны измениться, отправить ответ, запросить и отобразить обновленные данные;
– добавить кнопки, скачивающие файл с данными в форматах XML/HTML/JSON.

12 Лабораторная работа № 12. Разработка web-сервера

Цель работы: освоить работу с РСУБД, понять принципы дизайна RESTful API, получить навыки тестирования, валидации и документирования API.

Теоретический материал

REST API: для чего нужен и как работает (https://cloud.yandex.ru/ docs/ glossary/rest-api).

REST API (https://blog.skillfactory.ru/glossary/rest-api/).

СУБД PostgreSQL: почему ее стоит выбрать для работы с данными и как установить (https://practicum.yandex.ru/blog/chto-takoe-subd-postgresql/).

Графический клиент pgAdmin (https://metanit.com/sql/postgresql/1.2.php).

Инсталляция NongoDB-сервера (https://www.mongodb.com/try/download/ community).

Node.js-клиент для mongodb (https://www.npmjs.com/package/mongodb).

Задание для самостоятельного выполнения.

Создать проект ishop (интернет-магазин).

Тип приложения – SPA.

База данных: PostgreSQL/MongoDB.

Количество таблиц – не менее 5.

Создать CRUD API для работы с БД.

API разработать в соответствии с RESTful-архитектурой.

Протестировать работу API с использованием инструмента Postman API platform.

13 Лабораторная работа № 13. Работа с базой данных. ORM

Цель работы: научиться использовать ORM-библиотеки для приложений на Node.js.

Теоретический материал

Гибкая ORM для Node.js – Sequelize (https://proglib.io/p/gibkaya-orm-dlya-node-js-sequelize-2022-10-12).

Руководство по Sequelize (https://my-js.netlify.app/docs/guide/sequelize/).

Пакет mongoose (https://www.npmjs.com/package/mongoose).

Документация по mongoose (https://mongoosejs.com/docs/).

Задание для самостоятельного выполнения.

Выполняются все требования лабораторной работы № 12.

Создать модель БД с использованием ORM sequelize/mongoose;

Добавить валидацию на уровне сущностей при работе через ORM.

Разработать клиентскую часть приложения.

Требования к приложению

Дизайн – на ваше усмотрение. Как минимум название магазина, карточки товара, формы редактирования и добавления товара должны быть стилизованы.

Тематика сайта – на ваше усмотрение.

Адаптивная или отзывчивая верстка.

При разработке компонентов максимально использовать методы работы с массивами (forEach, map, filter и т. д.).

Список товаров в интернет-магазине должен хранить по каждому товару как минимум название, цену, URL-фотографию, сколько единиц товара осталось на складе.

Перечень товаров хранится в JSON-файле.

Сведения о товаре отображаются в виде карточек.

Карточка одного из товаров может быть сделана «выбранной» щелчком в любое место. Карточка выбранного товара выделяется цветом.

В каждой карточке товара есть кнопки «удалить» и «редактировать».

Под карточками товаров располагается кнопка «новый».

При нажатии на кнопку «удалить» у пользователя запрашивается подтверждения удаления (confirm) и товар удаляется. Можно удалить любой товар, не обязательно выделенный; после удаления товара, выделенный товар остается выделенным. Если удалялся выделенный товар, то выделения быть не должно.

При нажатии на кнопку «редактировать» карточка товара выделяется и товар переходит в режим редактирования с кнопками «сохранить» и «отмена»:

– при любых изменениях сразу запрещаются все кнопки «редактировать», «удалить», «новый»;

– при любых изменениях полей валидируется правильное заполнение полей (по любым правилам); сообщения об ошибках отображаются возле неправильно заполненных полей;

– при невалидном заполнении полей кнопка «сохранить» недоступна;

– если кликнуть на другую строку – должен включиться режим просмотра карточки этого товара (если в редактируемую сейчас карточку не были внесены изменения, иначе клик игнорируется);

– если кликнуть на кнопку «редактировать» другого товара, сразу включается редактирование карточки этого товара (если в редактируемую сейчас карточку не были внесены изменения, иначе нажатие кнопки игнорируется либо кнопка запрещается).

При нажатии на кнопку «новый» карточка товара переходит в режим добавления (пустая форма) с кнопками «добавить» и «отмена»:

– валидация работает аналогично;

– клики по карточкам товаров не должны ничего делать;

– выделенной цветом карточки товара не должно быть;

– все кнопки «удалить», «редактировать» и «новый» должны быть запрещены.

Список литературы

1 Figma – где скачать и как установить программу на компьютер? [Электронный ресурс]. – Режим доступа: https://web-design.center/ustanovka-figma/. – Дата доступа: 20.09.2023.

2 **Окунев**, **А.** Руководство по Figma [Электронный ресурс]. – Режим доступа: https://medium.com/slashdesigner/figma-guide-5235b8a8ab4f. – Дата доступа: 20.09.2023.

3 **Нагаева, И. А.** Основы web-дизайна. Методика проектирования: учебное пособие / И. А. Нагаева, А. Б. Фролов, И. А. Кузнецов. – Москва; Берлин: Директ-Медиа, 2021. – 236 с.

4 Figma с нуля – основы работы с Фигмой для веб-разработчика, верстальщика и дизайнера. Полный обзор [Электронный ресурс]. – Режим доступа: https://www.youtube.com/watch?v=z6mlqOGmjQQ. – Дата доступа: 20.09.2023.

5 Лекция 3: Основы разработки интерфейсов мобильных приложений интерфейсов [Электронный ресурс]. – Режим доступа: https://intuit.ru/studies/courses/12643/1191/lecture/21986?page=1. – Дата доступа: 20.09.2023.

6 Типы адаптивных макетов [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/158703/. – Дата доступа: 20.09.2023.

7 **Черный, Б.** Профессиональный TypeScript. Разработка масштабируемых JavaScript-приложений / Б. Черный. – Санкт-Петербург: Питер, 2022. – 352 с.

Приложение А
(справочное)

Варианты заданий для выполнения лабораторных работ

1 Автострахование.
2 Агентство по сдаче автомобилей в аренду.
3 Аренда коньков, роликов, велосипедов, лыж.
4 Аэропорт: пассажирское расписание и перевозки.
5 Банковская система вкладов (физических и юридических лиц).
6 Банковская система кредитования (физических и юридических лиц).
7 Биллинг сотовой компании.
8 Ветеринарная лечебница.
9 Клуб обучения танцам.
10 Магазин косметики.
11 Машиностроительное предприятие: система по разработке и модификации изделий (ведение архива, стандартов и пр.).
12 Нефтеперерабатывающая компания.
13 Парикмахерская.
14 Поставка вин.
15 Приемная комиссия вуза.
16 Производство мебели (прием индивидуальных и типовых заказов и изготовление).
17 Рекламное агентство.
18 Риелторская компания: аренда, продажа первичного и вторичного жилья.
19 Санаторий.
20 Система управления проектом для IT-компании.
21 Складская логистика.
22 Спа-салон (услуги, обслуживающий персонал и пр.).
23 Страховая компания.
24 Такси.
25 Транспортная логистика.
26 Туристическое агентство (путешествия за рубеж).
27 Туристическое агентство (путешествия по Беларуси).
28 Учет оборудования на крупном промышленном предприятии.
29 Филармония.
30 Электронный проездной.