ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ГОРОДА МОСКВЫ ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ ЦЕНТР ПРОФЕССИОНАЛЬНЫХ КВАЛИФИКАЦИЙ И СОДЕЙСТВИЯ ТРУДОУСТРОЙСТВУ «ПРОФЕССИОНАЛ»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к итоговой аттестационной работе на тему

«Разработка web-ресурса с использованием технологий HTML, CSS, JavaScript, SCSS, React, PHP»

на примере web-ресурса: https://react.bohohome.ru/

слушателя **Агафонова Антона Сергеевича** группы **№0358**

программы профессиональной переподготовки

«Frontend разработка»

Москва, 2023

Оглавление

ПОСТАНОВКА ЗАДАЧИ И ПЛАН РАБОТЫ 3

НАЗНАЧЕНИЕ ВЕБ-РЕСУРСА 4

СОЗДАНИЕ ПРОТОТИПА ВЕБ-РЕСУРСА 4

ЗАПУСК ИСХОДНОГО ВЕБ-РЕСУРСА 4

МАРШРУТИЗАЦИЯ 5

ОПИСАНИЕ СТРАНИЦ 5

ОПИСАНИЕ СИСТЕМЫ АВТОРИЗАЦИИ ПОЛЬЗОВАТЕЛЯ 6

НЕМНОГО О BACK-END 7

ОПИСАНИЕ РАБОТЫ ЛИЧНОГО КАБИНЕТА ПОЛЬЗОВАТЕЛЯ 7

РАЗМЕЩЕНИЕ ИСХОДНОГО ПРОГРАММНОГО КОДА 8

СПИСОК ЛИТЕРАТУРЫ 9

ПОСТАНОВКА ЗАДАЧИ И ПЛАН РАБОТЫ.

Задачей является создание веб ресурса по продаже товаров свободного назначения на базе темы Aroma Shop Landing Page. Основная задача работы – создание системы регистрации и авторизации пользователя на вышеуказанном сайте с личным кабинетом. В личном кабинете пользователь может менять свои данные, включая фото, выводить список пользователей, добавлять в друзья, выводить список друзей и информацию об отдельных пользователях.

Назначение веб-ресурса

Веб-ресурс предназначен для предоставления информации потенциальным клиентам о компании, ее продукции, ценах и т.д. Также для действующих клиентов разработаны система авторизации и личный кабинет пользователя. Информация об авторизации сохраняется в течение недели после авторизации, если пользователь не выйдет из системы.

Предусмотрены следующие разделы сайта:

Главная страница (home) – включает в себя баннер компании, каталог продукции, раздел новостей;

Магазин (shop) – содержит каталог продукции;

Страницы регистрации и авторизации (login) – с редиректом в личный кабинет при успешной авторизации;

Контакты (contact) – страница контактов с картой.

Создание прототипа веб-ресурса

Прототип создавался на базе готовой темы Aroma Shop Landing Page. В соответствии с выбранной темой был выполнен эскиз главной страницы.

Запуск исходного веб-ресурса

Фронтенд написан с использованием библиотеки React. Для этого установлена библиотека Create React App. Также в проекте использованы npm-пакеты react-router-dom для маршрутизации, node-sass для работы со стилями, axios для выполнения HTTP-запросов.

Изначально тема разработана с использованием фреймворка Bootstrap ver 4.6. Перед началом работы над проектом подключен этот фреймворк.

Маршрутизация

Маршруты написаны в файле AppRouter.js, на который ссылается исходный файл приложения App.js.

```
return (
  <>
    <Routes>
      <Route path="/" element={<Layout />}>
        <Route index element={<Main products={products} />} />
        <Route path="shop" element={<Shop products={products} />} />
        <Route path="contact" element={<Contacts />} />
        <Route path="register" element={<Register />} />
        <Route
          path="auth"
          element={
            <RequireAuth>
              <Login />
            </RequireAuth>
          }
        />
        <Route path="*" element={<NotFound />} />
      </Route>
      <Route
        path="/users/*"
        element={
          <RequireAuth>
            <Users />
          </RequireAuth>
        }
      />
    </Routes>
  </>
```

В элементе Layout содержатся Header и Footer. В Layout обернуты все роуты, кроме личного кабинета. В роутинге, помимо стандартных роутов, присутствуют два приватных («auth» - авторизация, «users» - личный кабинет). Переход по этим маршрутам зависит от того, авторизован ли пользователь.

Описание страниц

На главной странице расположены несколько статических баннеров и каталог произвольных продуктов (с фото, названием, категориями и ценами). Продукты получены из созданной ранее базы данных.

На странице shop – то же самое, но без баннеров.

На страницах по маршрутам «register» и «auth» – формы регистрации и авторизации пользователя.

На странице contact – контакты и карта из API yandex maps.

Описание системы авторизации пользователя

Для обеспечения приватности двух роутов создана обертка, которая называется RequireAuth.js. Это hoc (компонент высшего порядка), который принимает компонент, завернутый в него, делает внутри себя проверки и возвращает либо тот же компонент, либо редирект на другую страницу. Если пользователь переходит на страницу личного кабинета по маршруту «/users», то, если пользователь авторизован, переход происходит, иначе он направляется на страницу авторизации. Если пользователь переходит на страницу авторизации по маршруту «/auth», то, если пользователь авторизован, он перенаправляется на страницу личного кабинета.

Для эффективного получения и, возможно, изменения данных авторизованного пользователя из разных частей сайта можно использовать state manager, например, redux. В данном приложении использован встроенный метод React Context. Контекст прописан в файле AuthProvider.js. Это также hoc, который является провайдером контекста, и в который обернуто все приложение. Таким образом, из всех компонентов можно получить данные из свойств этого провайдера. Свойством провайдера является состояние – объект user и методы signin и signout (объявлены внутри AuthProvider.js). Для удобства создан хук useAuth.js, который вызывает встроенный хук useContext для получения данных из контекста.

Таким образом, можно получить данные авторизованного пользователя от сервера и поместить их в состояние user.

User является объектом вида:

{id: "id пользователя",

name: "имя",

lastname: "фамилия",

email: "e-mail",

foto: "название файла фото пользователя с расширением",

friends: "перечисление id пользователей-друзей"}

При авторизации отправляется запрос на сервер, и, если проверка e-mail и хеш пароля прошла успешно, то открывается сессия, в которую записывается id пользователя, также создается сессионная cookie, которая живет неделю. На стороне фронтенда данные пользователя с помощью хука useAuth записываются в состояние user и происходит переадресация на страницу личного кабинета.

При входе на сайт в файле AppRouter.js и в RequireAuth.js происходит проверка, если сессия существует, то из базы данных получается информация о пользователе и записывается в состояние user. Исходя из этого состояния изменяется в том числе внешний вид кнопок и иконок в шапке сайта.

Немного о Back-end

Back-end написан на PHP. Файл test.php принимает запросы от браузера, проверяет значение переменной postName, которую содержит любой запрос, и в зависимости от неё вызывает один из статических методов класса User.php. Есть методы регистрации, авторизации, получения данных о пользователе и всех пользователях, выхода из аккаунта, изменения данных пользователя, изменения или удаления фото пользователя, добавления друга.

Файл db.php содержит информацию о подключении к базе данных в переменной $mysqli. Для взаимодействия с базой данных используются sql запросы SELECT, INSERT, UPDATE.

Фото пользователей перезаписываются на сервер с помощью методов PHP.

Многие из методов класса User.php используются при работе в личном кабинете пользователя.

Описание работы личного кабинета пользователя

В файле страницы личного кабинета Users.js реализованы ссылки на профиль пользователя, список всех пользователей, список друзей, кнопка выхода из аккаунта, ссылка на главную страницу. В том же файле содержится

маршрутизация, описывающая переходы по ссылкам на профиль, список пользователей, друзей, плюс переход на страницу отдельного пользователя.

На странице профиля для загрузки фото используется input type="file". При замене фото происходит проверка размера (файл должен быть не более 2мб), в User.php проверка расширения файла (допустимы .jpg, .jpeg, .png, .gif, .webp), отправка данных о новом фото в базу данных, запись файла фото на сервер, в react изменение состояния user из контекста. После записи фото пользователя в базу данных появляется кнопка удаления фото, при нажатии на которую удаляется каталог с файлом фото и информация о названии файла из базы данных. Для изменения данных пользователя (имя, фамилия, e-mail) текстовые поля заменяются инпутами формы. При нажатии кнопку «Отправить» перезаписываются поля имя, фамилия, e-mail в базе данных и изменяется состояния user.

На странице списка пользователей происходит получение данных обо всех пользователях и загрузка их в виде таблицы. Там же можно добавить пользователя в друзья. В случае, если вы находитесь в списке друзей у другого пользователя, под его именем появляется соответствующая надпись. При добавлении новый список друзей пользователя отправляется в базу данных, обновляется состояние user.

Список друзей формируется на странице друзей и записывается в тот же компонент с таблицей, где так же можно удалять друзей из списка.

При нажатии кнопки выхода из аккаунта происходит сброс php-сессии и редирект на страницу авторизации.

Размещение исходного программного кода

Исходный код веб-ресурса размещен в репозитории GitHub по ссылке https://github.com/nappelbaum/react-app.

СПИСОК ЛИТЕРАТУРЫ

1. Документация по библиотеке React на русском языке [Электронный ресурс]: сайт. URL: https://ru.legacy.reactjs.org/

2. Документация по фреймворку Bootstrap на русском языке [Электронный ресурс]: офиц. сайт. URL: https://bootstrap-4.ru/

3. Официальная документация Яндекс по подключению API [Электронный ресурс]: офиц. сайт. URL: https://yandex.ru/dev/maps/jsapi/doc/2.1/quick-start/index.html

4. Официальная документация ресурса Fontawesome [Электронный ресурс]: офиц. сайт. URL: https://fontawesome.com/

5. Электронный учебник по HTML и CSS [Электронный ресурс]: офиц. сайт. http://htmlbook.ru/html5

6. Официальная документация Mozilla [Электронный ресурс]: офиц. сайт. URL: https://developer.mozilla.org/ru/