Публикация проекта

Оглавление

Вынести в отдельный файл путь к json-серверу из fetch.

Создадим папку src/helpers. В ней - файл variables.js:

```
export const serverPath = "http://localhost:8000/";
```

В файлах проекта сделаем импорт и подстановки:

```
import {serverPath} from "../../helpers/variables";
...
  fetch(serverPath + "products")
    .then((res) => res.json())
    .then((data) => {
      setCart(data);
    });
```

Серверная часть проекта на glitch.me

1. Отдельный проект cart-server с json-server.

На рабочем столе создадим папку **cart-server.** В консоли
npm init --yes
npm i json-server

В корне проекта создадим db.json и добавим туда данные из проекта.

В корне проекта создадим server.js, который будет запускать json-server:

```
const jsonServer = require("json-server");
const server = jsonServer.create();
const router = jsonServer.router("db.json");
const middlewares = jsonServer.defaults();
```

```
const port = 8000;

server.use(middlewares);
server.use(router);

server.listen(port);
```

Сделаем так, чтобы json server запускался со стандартной командой **npm run start.** Для этого добавим в package.json:

```
 "scripts": {
   "start": "node ./server.js",
   "test": "echo \"Error: no test specified\" && exit 1"
 },
```

Таким образом сервер json можно запустить:

npm run start

или

node ./server.js

Проверка запуска: в браузере **localhost:8000**

2. Репозиторий для cart-server на GitHub

Создадим на GitHub публичный репозиторий cart-server**.** Закинем туда **файлы из нашей локальной папки**, кроме node_modules, сделаем первый коммит.

3. Деплой и запуск cart-server на glitch

Зайти на сайт glitch.com и зарегистрироваться на нем (я зарегистрирована под профилем GitHub).

Далее создаем проект
New project
Import from GitHub

В строке набрать
https://github.com/glaznad/cart-server

Проверка работы: внизу **PREVIEW - Prewiew in new window**

Возьмем оттуда адрес проекта из адресной строки:

https://eminent-festive-viscose.glitch.me/

Публикация проекта на GitHub Pages

1. cart-react с API json-server на glitch, вместо localhost
Открываем наш локальный проект с корзиной. В файле variables.js меняем адрес на адрес с glitch:

```
export const serverPath = "https://eminent-festive-viscose.glitch.me/";
```

Протестировать!

2. Сборка build версии и правка путей в inde.html
В консоли в проекте сделаем сборку build:
npm run build

Сборка находится в папке build/
Нам нужно подправить пути в index.html:
Ctrl-S - файл будет отформатирован, и с ним можно работать:
Правим все пути, начинающиеся с **/** на ./

3. Публикация build версии в cart-build на GitHub

Создадим на GitHub публичный репозиторий cart-build**.** Закинем туда **файлы из нашей локальной папки build**, сделаем первый коммит.

4. Включение GitHub pages и проверка приложения

В настройках проекта на Github:

Settings - Pages - Source - Branch:main - Save

Появится ссылка https://glaznad.github.io/cart-build/ . Через некоторое время проверяем, тестируем проект.

Связанные репозитории в Github

В списке репозиториев найти справа **Customize your pins**

Публикация проекта на GitHub Pages с использованием gh-pages

С помощью этого инструмента из любого репозитория GitHub можно развернуть статический веб-сайт. Поддерживается Jekyll, доступен 1 бесплатный домен 3 уровня, SSL, неограниченный трафик.

1. В **консоли в проекте** устанавливаем:
npm install gh-pages --dev

2. В **Github** создаем новый репозиторий. Скопируем оттуда ссылку вида https://github.com/glaznad/euroflex.git

3. В **package.json:**
"homepage":https://nameaccount.github.io/namerepository //обратить внимание на ссылку!

Там же в разделе “scripts”:

```
"predeploy": "npm run build",
"deploy": "gh-pages -d build"
```

4. В **консоли в проекте**:
git init
git add -A
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/glaznad/euroflex.git
git push -u origin main

5. **В настройках проекта на Github**:
Settings - Pages - Source - Branch:gh-pages **- Save**

Получим ссылку

5. 1. После изменений в проекте в **консоли в проекте** устанавливаем:
npm run predeploy - только для проверки, можно не использовать
npm run deploy

Версия build будет храниться в **ветке gh-pages**

Если приложение использует **сервер на node.js,** то такой способ не подойдет

Публикация проекта с react-router-dom на GitHub Pages

Вместо импорта **BrowserRouter** необходимо импортировать HashRouter.

```
...
import { HashRouter as Router, Routes, Route } from "react-router-dom";

function App() {
...
 return (
   <Router>
    <div className="app">
     <Navbar />
```

```
      <Routes>
       <Route path="/" element={<FormPage optionsProducts={optionsProducts} />} />
       <Route
        path="/table"   // Пути страниц обязательно начинать с /
        element={
         <TablePage
          statuses={statuses}
          products={products}
          optionsProducts={optionsProducts}
         />
        }
       />
       <Route
        path="/edit/:id"
        element={
         <EditPage optionsProducts={optionsProducts} optionsStatuses={optionsStatuses} />
        }
       />
       <Route path="*" element={<NotFound />} />
      </Routes>
     </div>
    </Router>
  );
}
```

Если используются **параметры в строке поиска**, то нужно иметь ввиду, что они все будут в windows.location.hash, а не window.location.search

У меня проверка на наличие параметров сделана так:
```
if (window.location.hash.includes('?'))
```

Публикация проекта на Vercel

https://vercel.com/new?onboarding=true - сайт

Данный сервис позволяет собирать и размещать статические веб-сайты на различных фреймворках (поддерживаются как JS-фреймворки, так и, например, генераторы статических сайтов - Hexo, Hugo, Jekyll и другие).

Вот что включает в себя бесплатный тариф:

- 50 пользовательских доменов
- 100 Гб файлового пространства
- 100 Гб ежемесячного трафика
- Неограниченное количество проектов
- CLI-интерфейс
- Serverless, CDN, CI/CD

Регистрация через Гитхаб.

Можно с Гитхаба импортировать репозиторий (без продакшн версии). Деплой Versrl сделает автоматически.

Add to Dashboard - Visit - приложение запустится. Возьмем из адресной строки ссылку на него, н-р:
https://react-currency-converter-pi.vercel.app/

Если были сделаны изменения в Гитхабе, то Versel понимает, сто нужно приложение пересобрать. Он делает их автоматически.

Публикация проекта на Netlify

Netlify – это одна из наиболее продвинутых платформ для веб-разработки, помогающая программистам публиковать проекты в сети. Этот сервис упрощает жизнь пользователям благодаря автоматизированным инструментам тестирования, сборки и размещения приложений и сайтов в интернете.

Netlify - прямой конкурент Vercel. Однако, кроме функций, которые предоставляет предыдущий сервис, тут на бесплатном тарифе присутствует:

- Обработка до 100 отправленных форм в месяц
- До 1000 авторизованных пользователей в месяц

- Аналитика работы сайта

https://www.netlify.com/ - сайт

Регистрация через Гитхаб. Далее все аналогично предыдущему сервису. Можно с Гитхаба импортировать репозиторий (без продакшн версии) и сделать деплой. Получим ссылку

У меня ошибка при создании build версии

Аналогично, если были сделаны изменения в Гитхабе, то Versel понимает, сто нужно приложение пересобрать. Он делает их автоматически.

Публикация проекта на Heroku

Heroku позволяет запускать Full Stack приложения в контейнерах (так называемых Dynos). Поддерживается большое число языков программирования и фреймворков. Главный недостаток - после получаса бездействия проекты, размещенные на бесплатном тарифе, "засыпают", а повторный запуск контейнера требует определенного времени.

На стартовом тарифе доступны:

- 550 часов/месяц работы Dynos (1000 после привязки банковской карты)
- 512 Мб ОЗУ на 1 контейнер
- CI/CD, CLI
- По 1 домену на 1 контейнер

Попробуем залить проект с серверной частью json-server:

Блокирует новые аккаунты из РФ!

Публикация проекта на Surge

Хостинг статических сайтов. Бесплатно доступны:

- 1 сайт с 1 доменом третьего уровня
- Неограниченное количество сборок

https://surge.sh/ - сайт

В любом терминале

npm install --global surge
surge --help - смотрим команды

```
  surge whoami       show who you are logged in as
  surge logout       expire local token
  surge login        only performs authentication step
  surge list         list all domains you have access to
  surge teardown     tear down a published project
  surge plan         set account plan
```

Должны на своем компьютере собрать приложение и потом залить его

npm run build
cd build
surge
Далее нужно зарегистрироваться или войти (ввести почту и пароль - glaznad75@gmail.com)

Далее подтвердить проект. Потом выбрать домен, н-р:

react-currency-convertor.surge.sh

Далее формируется ссылка для доступа:
react-currency-convertor.surge.sh

Изменения делать вручную после build