05.02.2025

Решение по Тестовому заданию “tz2”

Задание №2

Тестирование функциональности веб-сайта «ROBOTX. Поиск и документирование багов.

Ниже привожу Тест-кейсы для тестирования указанного веб-сайта, а также инструкции по ручному и автоматизированному тестированию.

ПРИМЕЧАНИЕ: для иллюстрации процесса автоматизированного тестирования, мною создан проект, он называется так:

250205_01_60_TEST_TestirovanieRobotx-project_Optimakros

Проект сохранён в репозитории GitHub

Ссылка на проект: https://github.com/Vladislav-NewJoined/250205_01_60_TEST_TestirovanieRobotx-project_Optimakros.git

“Точкой входа” в проект, т.е. файлом с функционалом запуска, является файл “test.js” в папке “tests”, которая находится в корневой папке проекта.

Тест-кейсы для функционала входа в систему

1. Тест-кейс: Успешный вход с корректными данными

- **Предусловие:** Пользователь зарегистрирован с корректным email и паролем.
- **Описание:** Убедиться, что главная страница загружается без ошибок и отображает все элементы корректно.
- **Шаги:**
  1. Открыть страницу входа по ссылке https://login.company.com/username.
  2. Ввести корректный email.
  3. Ввести корректный пароль.
  4. Нажать на кнопку "Sign In".
- **Ожидаемый результат:** Пользователь успешно входит в систему и перенаправляется в Логин Центр.

2. Тест-кейс: Вход с некорректным email

- **Предусловие:** Пользователь зарегистрирован с корректным паролем.
- **Описание:** Проверить процесс регистрации нового пользователя.
- **Шаги:**
  1. Открыть страницу входа.
  2. Ввести некорректный email.
  3. Ввести корректный пароль.
  4. Нажать на кнопку "Sign In".
- **Ожидаемый результат:** Появляется сообщение об ошибке: “No user found for given credentials”.

3. Тест-кейс: Вход с некорректным паролем

- **Предусловие:** Пользователь зарегистрирован с корректным email.
- **Описание:** Убедиться, что зарегистрированный пользователь может войти в систему.
- **Шаги:**
  1. Открыть страницу входа.
  2. Ввести корректный email.
  3. Ввести некорректный пароль.
  4. Нажать на кнопку "Sign In".
- **Ожидаемый результат:** Появляется сообщение об ошибке: “No user found for given credentials”.

4. Тест-кейс: Вход с пустыми полями

- **Описание:** Проверить функциональность поиска на сайте.
- **Шаги:**
  1. Открыть страницу входа.
  2. Оставить поля email и пароль пустыми.
  3. Нажать на кнопку "Sign In".
- **Ожидаемый результат:** Появляется сообщение об ошибке о необходимости заполнения полей.

5. Тест-кейс: Превышение лимита неуспешных попыток входа

- **Предусловие:** Учетная запись пользователя активна.
- **Описание:** Убедиться, что пользователь может добавить товар в корзину.
- **Шаги:**
  1. Открыть страницу входа.
  2. Ввести некорректные данные (5 раз подряд).
  3. Нажать на кнопку "Sign In" после каждой попытки.
- **Ожидаемый результат:** После 5-й попытки появляется сообщение: “Too many failed sign in attempts. Please try again in 5 minutes.” Учетная запись блокируется на 5 минут.

6. Тест-кейс: Повторный ввод неверных данных после блокировки

- **Предусловие:** Учетная запись заблокирована на 5 минут.
- **Описание:** Убедиться, что при попытке входа с неверными данными после блокировки учетной записи, система корректно обрабатывает запрос и не позволяет пользователю войти.
- **Шаги:**
  1. Подождать 5 минут.
  2. Открыть страницу входа.
  3. Ввести некорректные данные (email и пароль).
  4. Нажать на кнопку "Sign In".

- **Ожидаемый результат:** Блокировка учетной записи происходит сразу, появляется сообщение об ошибке.

7. Тест-кейс: Опция "Do not remember me"

- **Предусловие:** Пользователь вошел в систему.
- **Описание:** Проверить, что при выборе опции "Do not remember me" сессия пользователя завершается автоматически после определенного времени бездействия, и при следующем доступе требуется повторный ввод учетных данных.
- **Шаги:**
  1. Выбрать опцию "Do not remember me".
  2. Подождать 5 минут бездействия.

- **Ожидаемый результат:** Сессия автоматически завершается, при попытке доступа к системе пользователь должен заново ввести логин и пароль.

8. Тест-кейс: Ссылка "Forgot password?"

- **Описание:** Убедиться, что ссылка "Forgot password?" на странице входа корректно перенаправляет пользователя на страницу восстановления пароля, позволяя ему восстановить доступ к учетной записи..
- **Шаги:**
  1. Открыть страницу входа.
  2. Нажать на ссылку "Forgot password?".
- **Ожидаемый результат:** Пользователь перенаправляется на страницу восстановления пароля.

Эти тест-кейсы помогут проверить основные функции входа в систему, а также обработку ошибок и поведение системы в различных сценариях. Если у вас есть дополнительные требования или вопросы, дайте знать!

****

Инструкция по ручному тестированию веб-сайта «ROBOTX»

1. Знакомимся с тест-кейсами и убеждаемся, что у нас есть доступ к веб-сайту «ROBOTX».
2. Следуем шагам, описанным в каждом тест-кейсе, фиксируя результаты и любые обнаруженные ошибки.
3. Используем инструменты разработчика в браузере для отладки и анализа ошибок (например, консоль JavaScript).
4. Заполняем отчет о тестировании с указанием выполненных тестов, найденных ошибок и их статуса

Рекомендации по ручному тестированию:

- Открываем файл index в браузере и проверяем:

- Корректность отображения всех элементов интерфейса.
- Работу всех кнопок, форм и ссылок.
- Обработку ошибок (например, ввод некорректных данных).
- Проверяем кросс-браузерную совместимость (Chrome, Firefox, Edge и т.д.).
- Проверяем адаптивность страницы (отображение на мобильных устройствах).

Инструкция по автоматизированному тестированию веб-сайта «ROBOTX» (я в тестовом тестировании использовал именно автоматизированное тестирование)

Для автоматизированного тестирования можно использовать такие инструменты, как **Selenium**, **Cypress** или **Jest** (я использовал Selenium). Каждый из них имеет свои особенности:

- **Selenium:** Подходит для тестирования веб-приложений на разных браузерах. Хорошо подходит для сложных сценариев.
- **Cypress:** Идеален для тестирования современных веб-приложений, обеспечивает быструю обратную связь и имеет простую настройку.
- **Jest:** Чаще используется для тестирования JavaScript-кода (например, React-приложений) и не является основным инструментом для тестирования интерфейса.

Для тестового тестирования устанавливаем с официальных сайтов следующие программные продукты:

- **Node.js**
- **Selenium WebDriver** (“Selenium WebDriver” и “ChromeDriver” устанавливаются в терминале (“cmd” или “PowerShell” при помощи команды: “npm install selenium-webdriver chromedriver”, - если у Вас Windows. Если Linux, то, как правило используется терминал “bash”, и команда может немного отличаться)
- **ChromeDriver** (если используется браузер Google Chrome)
- **Visual Studio Code** (редактор для написания скрипта тестов, устанавливается при необходимости)

- Далее, создаём в отдельной папке проект. (Я создал проект с названием: **250205_01_60_TEST_TestirovanieRobotx-project_Optimakros**

(ссылка на GitHub: https://github.com/Vladislav-NewJoined/250205_01_60_TEST_TestirovanieRobotx-project_Optimakros.git

- В корневой папке проекта создаём папку “tests”, в ней - файл “test.js”
- В файл “test.js” прописываем скрипт для тестового тестирования заголовка. Я прописал следующий скрипт:

```
const { Builder, By, until } = require('selenium-webdriver');

(async function test() {
  // Инициализация драйвера
```

```
  let driver = await new Builder().forBrowser('chrome').build();

  try {
    // Открытие сайта
    await driver.get('http://localhost:3000/');

    // Проверка заголовка страницы
    let title = await driver.getTitle();
    if (title.includes('ROBOTX')) {
      console.log('Тест пройден: заголовок страницы корректен.');
    } else {
      console.log('Тест не пройден: заголовок страницы некорректен.');
    }
  } finally {
    // Закрытие браузера
    await driver.quit();
  }
})();
```

На этом подготовка к тестированию закончена. Далее начинаем само тестирование. Для этого выполняем следующие шаги:

- **Инициализируем проект.** Для этого выполняем в терминале (я использовал терминал cmd (или Command Prompt), для Windows также можно использовать терминал PowerShell) последовательно следующие команды:
  - cd <путь к папке проекта>
  - npm init –y
  - npm start

По результату, веб сайт ROBOTX откроется по этому адресу: http://192.168.0.12:3000
а также по этому адресу: http://localhost:3000/
после этого, в папке проекта (т.е. после команды “cd <путь к папке проекта>”)

- **Запускаем скрипт в файле “test.js”.** Для этого нужно в проводнике Windows зайти в корневую папку проекта (либо с помощью команды в терминале cmd (Command Prompt) или PowerShel: «cd <путь к папке проекта>») и выполнить там команду «node tests/test.js», либо зайти в проводнике непосредственно в папку “tests”, где содержится файл «test.js», и выполнить там команду «node test.js». После этого запустится процесс тестирования, и по результату в терминале появится надпись: **«Тест пройден: Заголовок страницы корректен.»**.

Таким образом, мы протестировали заголовок страницы. Далее подобным образом мы можем протестировать и другие элементы веб сайта ROBOTX, например:

- Формы
- Кнопки
- Ссылки
- Навигация
- Изображения и медиа
- Таблицы
- Сообщения об ошибках
- Адаптивность
- Состояние сессии пользователя
- Производительность

С уважением, Созин Владислав
+7 909 328-59-88 (WhatsApp)
https://t.me/tess_SV (Telegram)
sozin.vladislav@mail.ru