ЛАБОРАТОРНАЯ РАБОТА №6

Тема лабораторной работы: автоматизация тестирования

**1. Разработка автотестов**

**a. Цель работы**

Цель работы — разработка автотестов для проверки функциональности приложения "Бронирование авиабилетов", учитывая свойства сопровождаемости, надежности и структурированности. Автотесты должны воспроизводить поведение пользователя при взаимодействии с элементами пользовательского интерфейса и обеспечивать контроль результатов выполненного сценария.

**b. Описание реализованных автотестов: инструменты, подходы**

**Инструменты:**

* **Formy Project**: Этот веб-ресурс предлагает разнообразные формы для тестирования, такие как форма регистрации, форма отправки файла и другие.
* **UITesting Playground**: Интерактивная песочница, предназначенная для тестирования пользовательских интерфейсов. Предоставляет различные компоненты и элементы управления для применения различных тестовых сценариев.
* **The Internet**: Веб-приложение, содержащее набор различных страниц и функций для тестирования веб-приложений. Предоставляет множество сценариев для проверки функциональности и интерактивных элементов.

**Подходы:**

**Модульное тестирование**: проверка функциональности отдельных компонентов и функций приложения.

**Тестирование пользовательского интерфейса** с использованием эмуляции поведения пользователя.

**Контроль результатов выполненных сценариев** через проверку ожидаемых результатов и утверждений.

**c. Код автотестов**

Код представляет набор автотестов для тестирования функциональности веб-приложения, связанного с бронированием авиабилетов. В этих тестах проверяется корректность работы следующих сценариев:

1. Добавление пассажира с валидными данными.
2. Попытка добавления пассажира с невалидным именем (пустая строка).
3. Попытка добавления пассажира с невалидной фамилией (пустая строка).

Каждый тест проверяет ожидаемый результат при выполнении соответствующего сценария и сравнивает его с фактическим результатом на веб-странице. Если ожидаемый и фактический результаты не совпадают, тест будет считаться неудачным.

Код для тестирования представлен в «Приложение 1».

**d. Отчет о тестировании**

Тест-кейс 1: Проверка ввода валидных данных для добавления пассажира

* Результат: Успешно
* Описание: Пассажир успешно добавлен в систему.

Тест-кейс 2: Проверка ввода невалидного имени пассажира

* Результат: Успешно
* Описание: Сообщение об ошибке появилось при попытке добавить пассажира с пустым именем.

Тест-кейс 3: Проверка ввода невалидной фамилии пассажира

* Результат: Успешно
* Описание: Сообщение об ошибке появилось при попытке добавить пассажира с пустой фамилией.

**e. Выводы по работе**

В результате тестирования приложения для бронирования авиабилетов была подтверждена его работоспособность и стабильность. Приложение успешно обрабатывает ввод данных о пассажирах и бронированиях, корректно реагирует на ошибочные вводы и предоставляет информативные сообщения об ошибках. Использование автоматизированных тестов позволило провести проверку функциональности более эффективно и ускорить процесс тестирования.

f. **Список использованных источников**

1. "React документация" - официальная документация React. URL: [https://react.dev](https://react.dev/)
2. "Node.js документация" - официальная документация Node.js. URL: <https://nodejs.org/en/docs/>
3. "MDN Web Docs" - ресурс Mozilla Developer Network. URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>

**Приложение 1**

import React from 'react';

import { render, fireEvent } from '@testing-library/react';

import '@testing-library/jest-dom/extend-expect';

import Booking from './Booking';

describe('Booking', () => {

it('adds a valid passenger', () => {

const { getByLabelText, getByText } = render(<Booking />);

const nameInput = getByLabelText('Имя пассажира');

const surnameInput = getByLabelText('Фамилия пассажира');

const addButton = getByText('Добавить');

fireEvent.change(nameInput, { target: { value: 'Иван' } });

fireEvent.change(surnameInput, { target: { value: 'Иванов' } });

fireEvent.click(addButton);

expect(getByText('Пассажир успешно добавлен в систему')).toBeInTheDocument();

});

it('displays error message for invalid name', () => {

const { getByLabelText, getByText } = render(<Booking />);

const nameInput = getByLabelText('Имя пассажира');

const surnameInput = getByLabelText('Фамилия пассажира');

const addButton = getByText('Добавить');

fireEvent.change(nameInput, { target: { value: '' } });

fireEvent.change(surnameInput, { target: { value: 'Иванов' } });

fireEvent.click(addButton);

expect(getByText('Введите имя пассажира')).toBeInTheDocument();

});

it('displays error message for invalid surname', () => {

const { getByLabelText, getByText } = render(<Booking />);

const nameInput = getByLabelText('Имя пассажира');

const surnameInput = getByLabelText('Фамилия пассажира');

const addButton = getByText('Добавить');

fireEvent.change(nameInput, { target: { value: 'Иван' } });

fireEvent.change(surnameInput, { target: { value: '' } });

fireEvent.click(addButton);

expect(getByText('Введите фамилию пассажира')).toBeInTheDocument();

});

});