Playwright Test Automation for SauceDemo

This project contains automated tests for the SauceDemo e-commerce website using Playwright, TypeScript, and the Page Object Model pattern.

Prerequisites

- Node.js (version 20 or higher)
- npm or yarn package manager

Setup Instructions

1. Clone the Repository or Extract the Zip File

If you have access to the repository:

```
git clone https://github.com/oludarebusari/TETakeHomeAssignment.git
cd TETakeHomeAssignment
```

If you received a zip file, extract it and navigate to the project directory:

```
# Extract the zip file (e.g., TETakeHomeAssignment.zip)
# Then navigate to the extracted directory
cd TETakeHomeAssignment
```

2. Install Dependencies

```
npm install
```

3. Environment Configuration

Create a `.env` file in the root directory with your SauceDemo credentials:

```
SAUCEDEMO_USERNAME=your_username
SAUCEDEMO_PASSWORD=your_password
```

Note: The `.env` file is already included in `.gitignore` to prevent committing sensitive information.

4. Install Playwright Browsers

```
npx playwright install
```

5. Run Tests

Run all tests

```
npx playwright test
```

Run specific test file

```
npx playwright test tests/ui/login.spec.ts
npx playwright test tests/ui/add_item.spec.ts
```

Run tests with UI mode

```
npx playwright test --ui
```

Run tests in headed mode (visible browser)

```
npx playwright test --headed
```

Generate and view HTML report

```
npx playwright show-report
```

Generate and serve Allure report

```
npm run allure:generate
npm run allure:serve
```

Project Structure

```
├── page-objects/           # Page Object Model classes
│   ├── loginPage.ts       # Login page interactions
```

```
│   └── dashboardPage.ts   # Dashboard/inventory page interactions
├── tests/
│   └── ui/
│       ├── login.spec.ts  # Login functionality tests
│       └── add_item.spec.ts # Cart functionality tests
├── fixtures/              # Test data fixtures
│   ├── loginData.ts       # Login credentials
│   └── cartItems.ts       # Cart items data
├── types.ts               # TypeScript interfaces
├── playwright.config.ts   # Playwright configuration
├── package.json           # Project dependencies
└── .env                   # Environment variables (not committed)
```

Test Coverage

Login Tests (`tests/ui/login.spec.ts`)

- Page title verification
- Successful login with valid credentials
- Error handling for invalid credentials
- Error handling for empty username
- Error handling for empty password

Add Item Tests (`tests/ui/add_item.spec.ts`)

- Cart badge visibility when empty
- Adding multiple items to cart
- Cart badge count validation

Key Features

- **Page Object Model**: Clean separation of test logic and page interactions
- **TypeScript**: Type safety and better developer experience
- **Environment Variables**: Secure credential management
- **Data-driven Tests**: Fixtures for reusable test data
- **Robust Locators**: Uses data-test attributes for reliable element selection
- **Comprehensive Assertions**: Validates both UI state and business logic

Configuration

The project uses Playwright's configuration in `playwright.config.ts` with:

- Base URL set to `https://www.saucedemo.com/`
- Chromium browser for testing
- HTML reporting enabled
- Screenshot and video capture on failure

Note: Some browser configurations (Firefox, WebKit, mobile browsers, and branded browsers) are commented out in the config file. This does not indicate incomplete code; these options were intentionally disabled as they are not required for this specific project, which focuses on testing with Chromium.

Best Practices Implemented

- Factory functions for page objects (createLoginPage and createDashboardPage)
- Input validation in page methods
- Descriptive test names and comments
- Proper error handling
- Separation of concerns between page objects and tests
- Environment variable usage for sensitive data