Software Testing TooL - Playwright

Step 1: Set Up the Project

1.1 Install Node.js (If Not Installed)

If you haven't installed Node.js, download and install it from:
🔗 https://nodejs.org/

To check if Node.js and npm are installed, run in cmd:

```
node -v
npm -v
```

This should display the installed versions.

1.2 Create a New Project

Open a terminal and run:

```
mkdir playwright-todo
cd playwright-todo
npm init -y
```

This initializes a new Node.js project with a package.json file.

1.3 Install Required Packages

```
npm install express body-parser cors
```

- express → Creates the backend server.
- body-parser → Parses incoming JSON requests.
- cors → Enables cross-origin requests (for frontend to talk to backend).

Step 2: Build the Backend

Create a file **server.js**

Step 3: Run the Backend

Start the server:

node server.js

Now, your backend is running at **http://localhost:3000**.

To check if the API is working properly.

Open a new terminal and run:

curl -X GET http://localhost:3000/todos

If everything is set up correctly, it should return an empty array: (in json)

[]

Next Step: Create the Frontend

We will now build a simple **HTML+JavaScript** frontend to interact with the backend.

1. Create an index.html File

2. Open the Frontend in a Browser

Now, open the index.html file in **any browser** (Chrome, Edge, or Firefox).
You should see a simple To-Do List where you can **add, mark as completed, and delete tasks**.

Now, let's set up **Playwright** for testing the to-do list application.

Step 1: Install Playwright

Open your terminal inside the **playwright-todo** project folder and run:

npm install -D @playwright/test

This installs Playwright for testing.

Step 2: Set Up Playwright Config (Optional)

Run the following command to initialize Playwright:

npx playwright install

This installs browsers for testing (Chromium, Firefox, and WebKit).

Step 3: Create a Playwright Test File

Inside your **playwright-todo** folder, create a new folder called tests and add a file named todo.test.js:
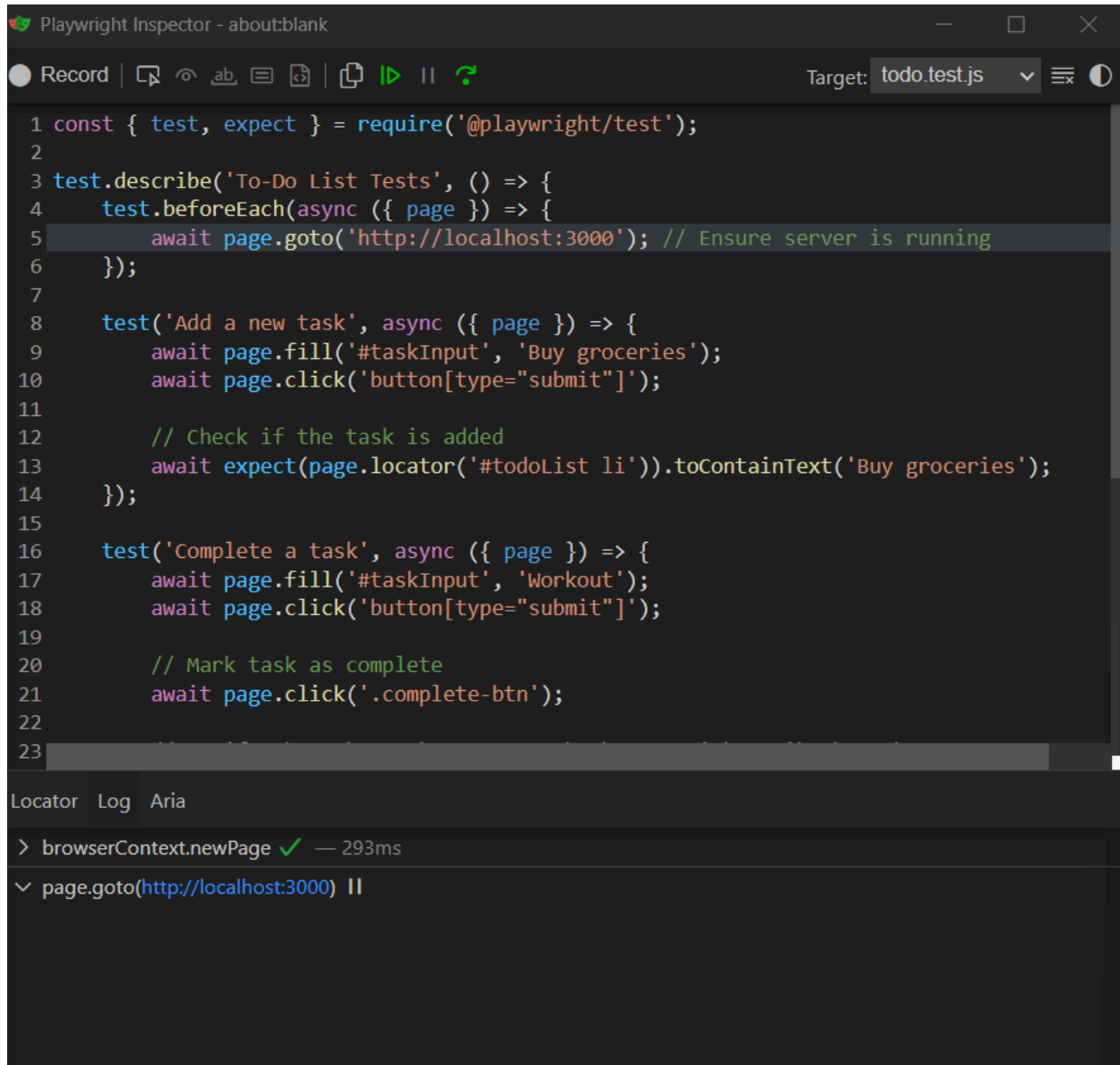
mkdir tests
cd tests
touch todo.test.js

Step 4: Write Test Cases in todo.test.js

Step 5: Run the Tests

Go back to your project root folder (playwright-todo) and run:

npx playwright test

This will **open a headless browser**, perform the tests, and show results.

```
1 const { test, expect } = require('@playwright/test');
2
3 test.describe('To-Do List Tests', () => {
4     test.beforeEach(async ({ page }) => {
5         await page.goto('http://localhost:3000'); // Ensure server is running
6     });
7
8     test('Add a new task', async ({ page }) => {
9         await page.fill('#taskInput', 'Buy groceries');
10        await page.click('button[type="submit"]');
11
12        // Check if the task is added
13        await expect(page.locator('#todoList li')).toContainText('Buy groceries');
14    });
15
16    test('Complete a task', async ({ page }) => {
17        await page.fill('#taskInput', 'Workout');
18        await page.click('button[type="submit"]');
19
20        // Mark task as complete
21        await page.click('.complete-btn');
22
23
```

Locator   Log   Aria

> browserContext.newPage ✓ — 293ms

∨ page.goto(http://localhost:3000) ‖

Step 6:To open Playwrite UI
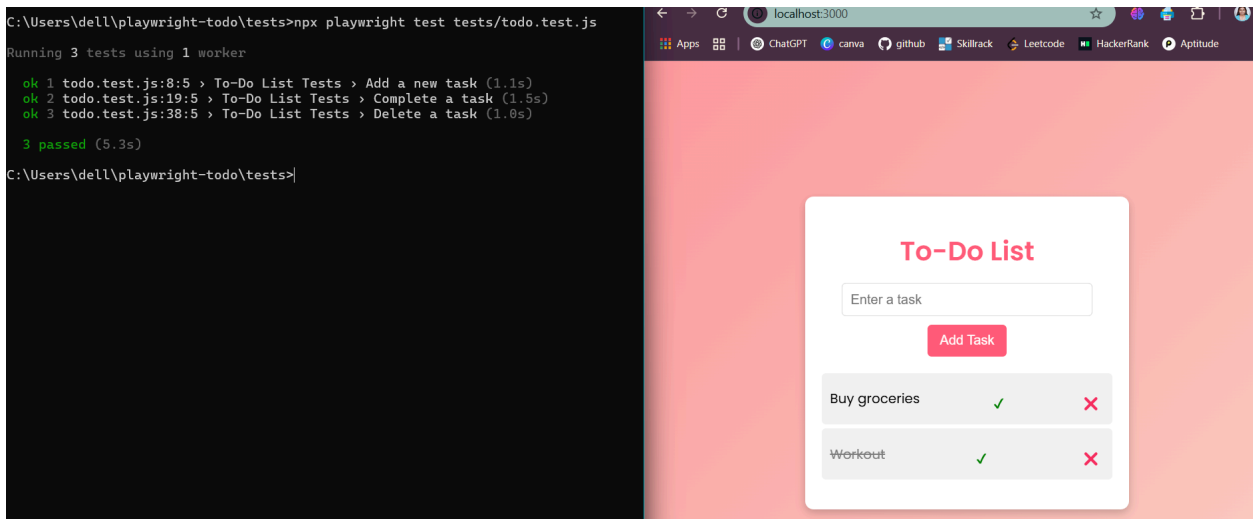
Go back to your project root folder (playwright-todo) and run:

npx playwright test --ui

Step 7:Then run the cmd and refresh often to check the test

Values will be updated automatically
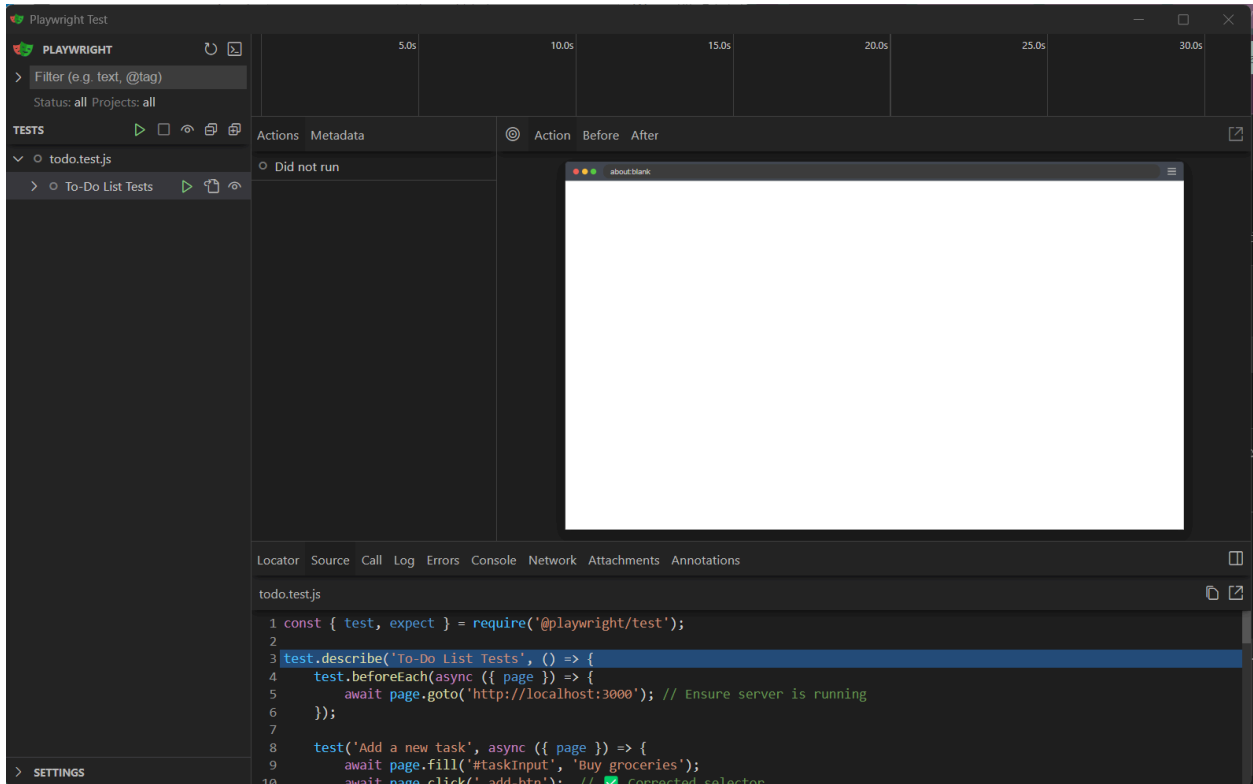
npx playwright test tests/todo.test.js

```
C:\Users\dell\playwright-todo\tests>npx playwright test tests/todo.test.js

Running 3 tests using 1 worker

  ok 1 todo.test.js:8:5 › To-Do List Tests › Add a new task (1.1s)
  ok 2 todo.test.js:19:5 › To-Do List Tests › Complete a task (1.5s)
  ok 3 todo.test.js:38:5 › To-Do List Tests › Delete a task (1.0s)

  3 passed (5.3s)
```