E2E Testing Environment & Planning

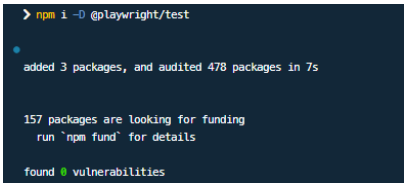
Phase 1: Environmental & Tool Selection

E2E Framework: Playwright

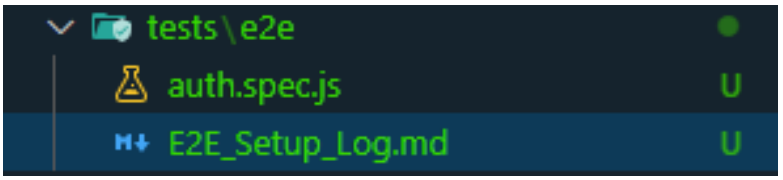
Install and Verify:

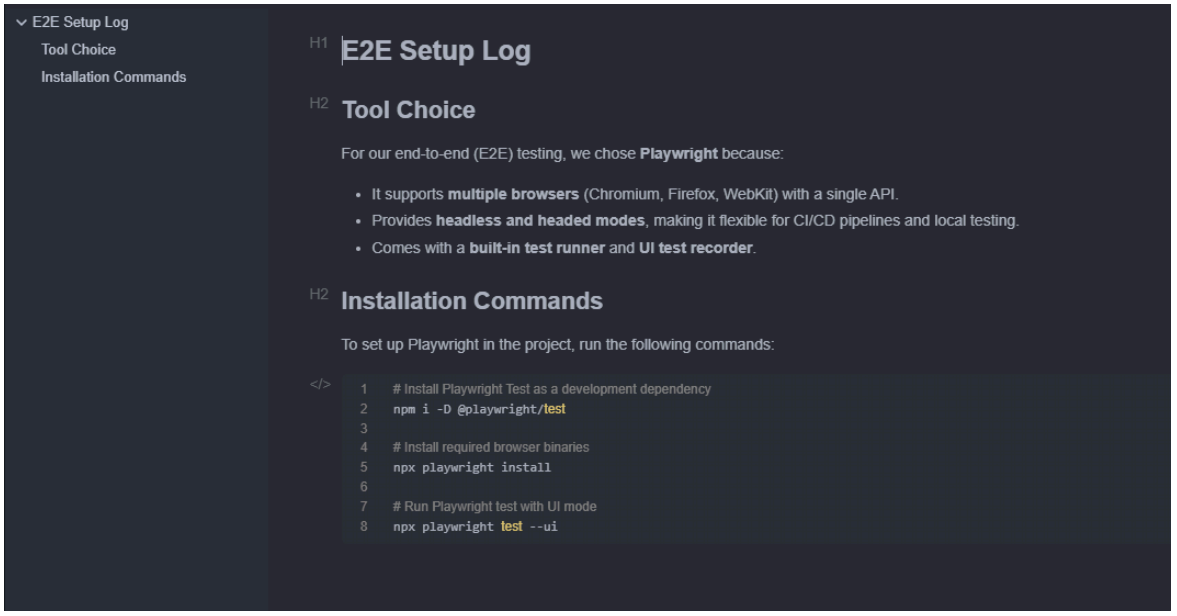
npm i -D @playwright/test
npx playwright install
npx playwright test --ui

Installation



Readme & folder





Phase 2: E2E Test Planning

Scenario ID: E2E-001
Title: User login and dashboard access
Preconditions: Google OAuth configured
Steps:
1) Open the home page
2) Click "Sign in with Google"
3) After callback, navigate to the dashboard
4) Verify user name/email visible
Expected: Dashboard shows authenticated user info

Scenario ID: E2E-002
Title: Search For Locations
Preconditions: User Logged in
Steps:
1) Open the map page
2) Click the "Search" bar at the top
3) Type in the location
4) Verify the specified location and show it on the map.
Expected: The map displays the user’s input location as a colored area.

Scenario ID: E2E-003
Title: Navigation Bar Selection
Preconditions: The User selected the region on the map.
Steps:
1) Click on the navigation bar at the top (scout, news, history)
2) After the input, the system shows a different page according to the button clicked.
3) Verify that the location card appears in the detail bar.
Expected: Expect the result of each mode to appear if available, and display 'Data Empty' if it is not.

Scenario ID: E2E-004
Title: Incorrect Method of Submission of The Location
Preconditions: The User opened the add location page
Steps:

1) Open the submission form.
2) Enter the information according to the given spaces.
3) If the information was entered in the incorrect format or left blank, the submission will not proceed.
4) Verify the user's input and show the validation message on display for correction.

Expected: The form displays messages for the user to input information in the correct format or ensure that every box is filled in.

Phase 3: Test Implementation

E2E (Playwright)
// tests/e2e/auth.spec.js import { test, expect } from '@playwright/test';

```
import fs from "fs";
import { test } from "@playwright/test";
import dotenv from "dotenv";
import { chromium } from "playwright";

// Load .env.local only if running locally
if (!process.env.GITHUB_ACTIONS) {
  dotenv.config({ path: ".env.local" });
}

test("Google Sign-In persistent login", async () => {
  const cookiePath = "/tmp/chromium-session"; // use /tmp for GitHub
Actions
  console.log("Step 1: Set cookie path:", cookiePath);

  // Remove previous session folder
  if (fs.existsSync(cookiePath)) {
    console.log("Step 2: Removing previous session folder");
    fs.rmSync(cookiePath, { recursive: true, force: true });
  } else {
    console.log("Step 2: No previous session folder found");
  }

  // Launch persistent browser context
  console.log("Step 3: Launching persistent browser context");
  const browser = await chromium.launchPersistentContext(cookiePath, {
    headless: false,
    args: [
```

```
      `--disable-blink-features=AutomationControlled`,
      "--no-sandbox",
      "--disable-setuid-sandbox",
    ],
  });

  const page = await browser.newPage();
  console.log("Step 4: Opened new page");

  // Go to your web app
  console.log("Step 5: Navigating to http://localhost:3000/");
  await page.goto("http://localhost:3000/");
  await page.screenshot({ path: "/tmp/after-goto.png" });
  console.log("Step 5: Screenshot taken after page load");

  // Click Google Sign-In button
  console.log('Step 6: Clicking "Sign In with Google" button');
  await page.getByRole("button", { name: "Sign In with Google"
}).click();

  // Perform Google login
  console.log("Step 7: Filling email");
  await page
    .getByRole("textbox", { name: "Email or phone" })
    .fill(process.env.GOOGLE_SIGNIN_EMAIL);
  await page.getByRole("button", { name: "Next" }).click();

  console.log("Step 8: Waiting for password input");
  await page
    .getByRole("textbox", { name: "Enter your password" })
    .fill(process.env.GOOGLE_SIGNIN_PASSWORD);
  await page.click("#passwordNext");
  console.log("Step 9: Filling password");

  // Wait for redirect to your app after login
  console.log("Step 10: Waiting for redirect to app");
  await page.waitForURL("http://localhost:3000/**");

  const screenshotPath = "/tmp/after-login.png";
  await page.screenshot({ path: screenshotPath, fullPage: true });
  console.log("✅ Step 10.1: Screenshot saved after redirect:",
screenshotPath);
```

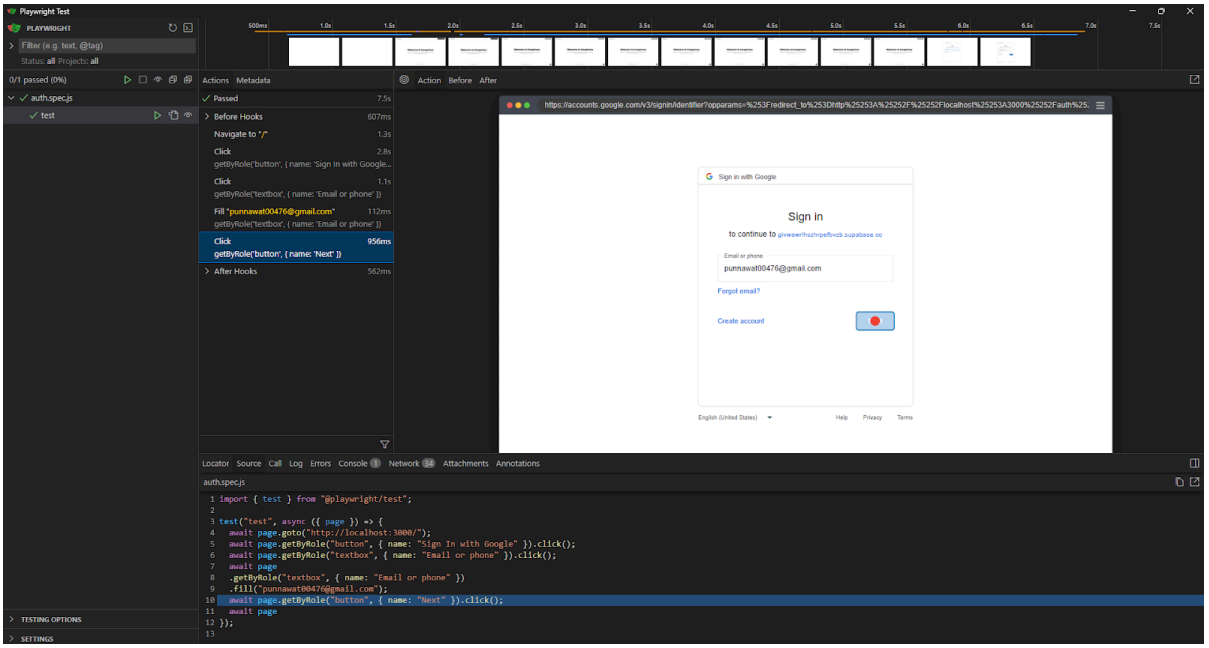
```
  await browser.close();
  console.log("Step 11: Browser closed, test complete");
});
```

```
> npx playwright test

Running 1 test using 1 worker

  ✓  1 tests\e2e\auth.spec.js:3:5 › test (7.2s)

1 passed (11.1s)
```



CI/CD Github Action Integration:

- yml file

```yaml
name: Playwright Google Sign-In Test

on:
  push:
    branches:
      - feat/testing
  pull_request:
    branches:
      - main

jobs:
  test:
    runs-on: ubuntu-latest

    env:
      # Secrets injected as environment variables
      GOOGLE_SIGNIN_EMAIL: ${{ secrets.GOOGLE_SIGNIN_EMAIL }}
      GOOGLE_SIGNIN_PASSWORD: ${{ secrets.GOOGLE_SIGNIN_PASSWORD }}
      NEXT_PUBLIC_SUPABASE_URL: ${{ secrets.NEXT_PUBLIC_SUPABASE_URL }}
      NEXT_PUBLIC_SUPABASE_PUBLISHABLE_OR_ANON_KEY: ${{
secrets.NEXT_PUBLIC_SUPABASE_PUBLISHABLE_OR_ANON_KEY }}

    steps:
      - name: Checkout repository
        uses: actions/checkout@v3

      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: 20

      - name: Install dependencies
        run: npm install

      - name: Install Xvfb
        run: sudo apt-get update && sudo apt-get install -y xvfb

      - name: Install Playwright browsers
        run: npx playwright install
```

```
      - name: Start local server
        run: |
          npm run dev &
          npx wait-on http://localhost:3000

      - name: Run Playwright tests in headed mode
        run: xvfb-run -a npx playwright test tests/e2e/ --headed
--workers=1

      - name: Upload screenshots
        if: always()
        uses: actions/upload-artifact@v4
        with:
          name: screenshots
          path: /tmp/*.png
```

Results of CI/CD: