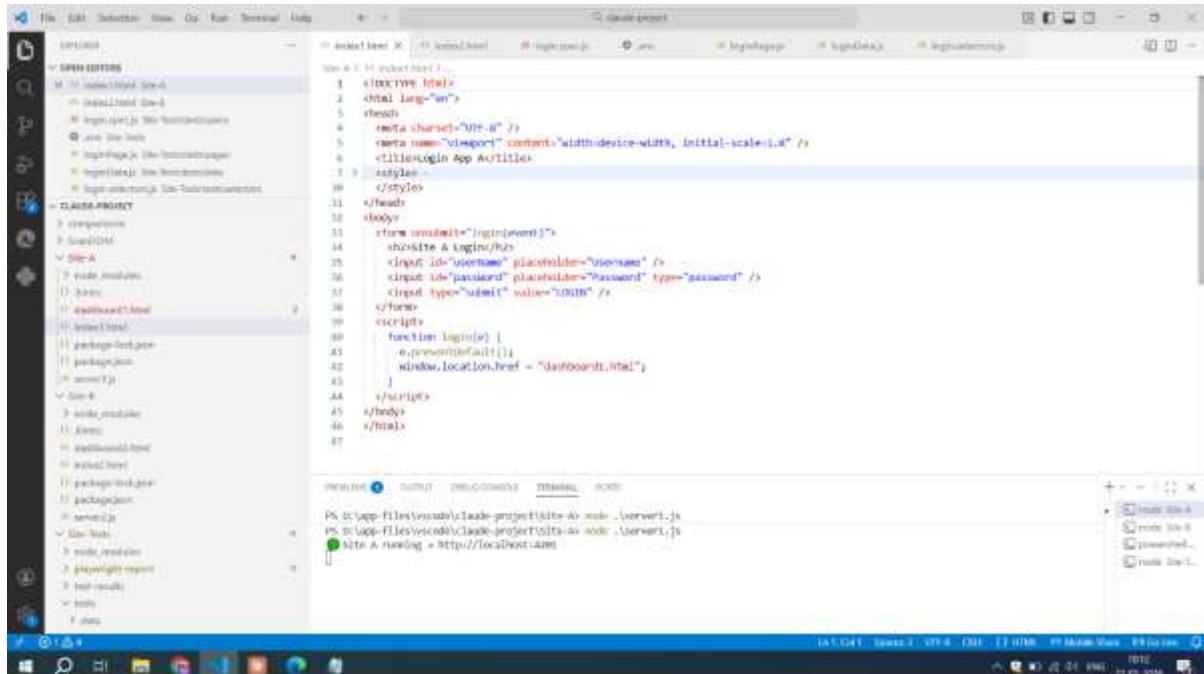


Documentation of the Project

Agentic UI testing

1 the websites – Baseline (SiteA) and Variant (SiteB)

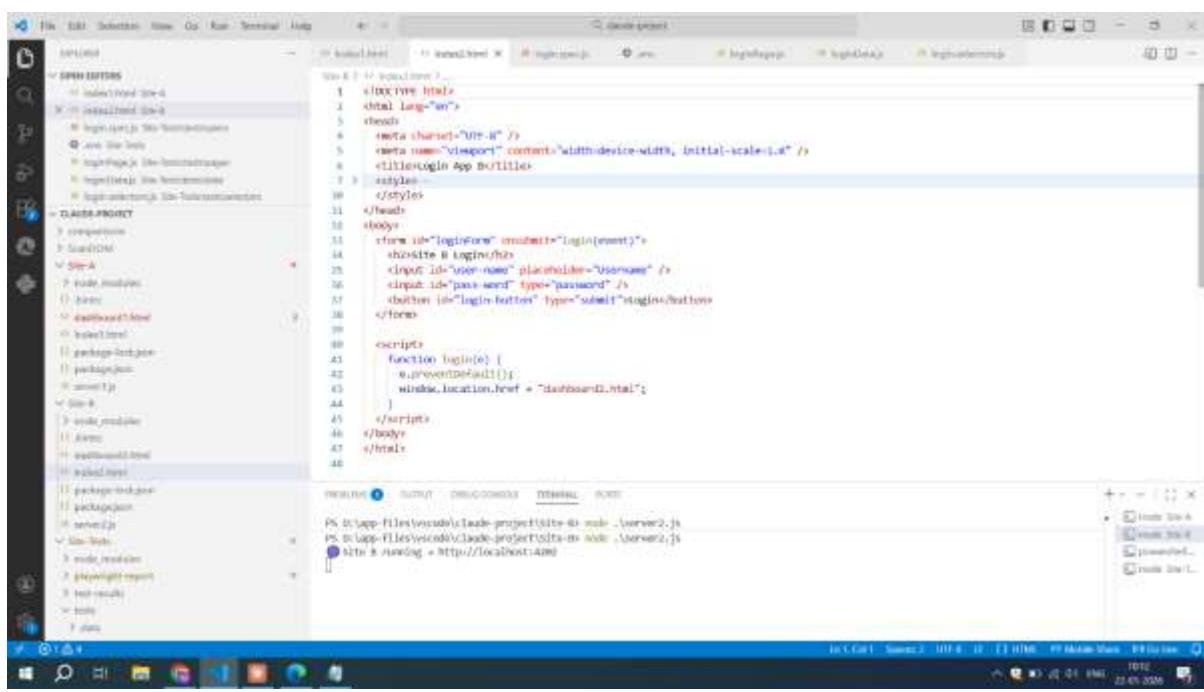


The screenshot shows a code editor with the file `index.html` open. The code is an HTML form for logging in:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Login App</title>
  <style>
    </style>
</head>
<body>
  <form id="loginForm" onsubmit="login(event)">
    <div>A login form</div>
    <input id="user-name" placeholder="username" />
    <input id="pass-word" placeholder="password" type="password" />
    <input type="submit" value="Login" />
  </form>
  <script>
    function login(e) {
      e.preventDefault();
      window.location.href = "dashboard.html";
    }
  </script>
</body>
</html>
```

Below the code, the terminal shows:

```
PS C:\Users\vinay\OneDrive\Projects\SiteA\siteA> node .\server.js
PS C:\Users\vinay\OneDrive\Projects\SiteA\siteA> node .\server.js
Site A is running at http://localhost:4000
```

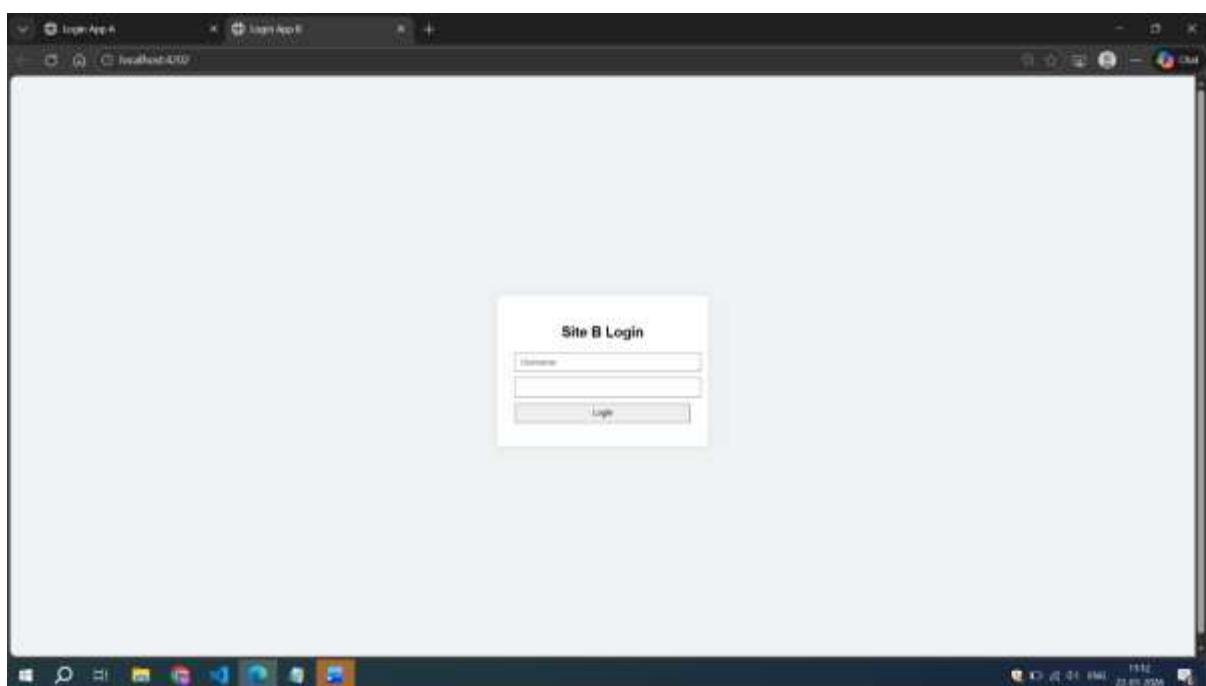
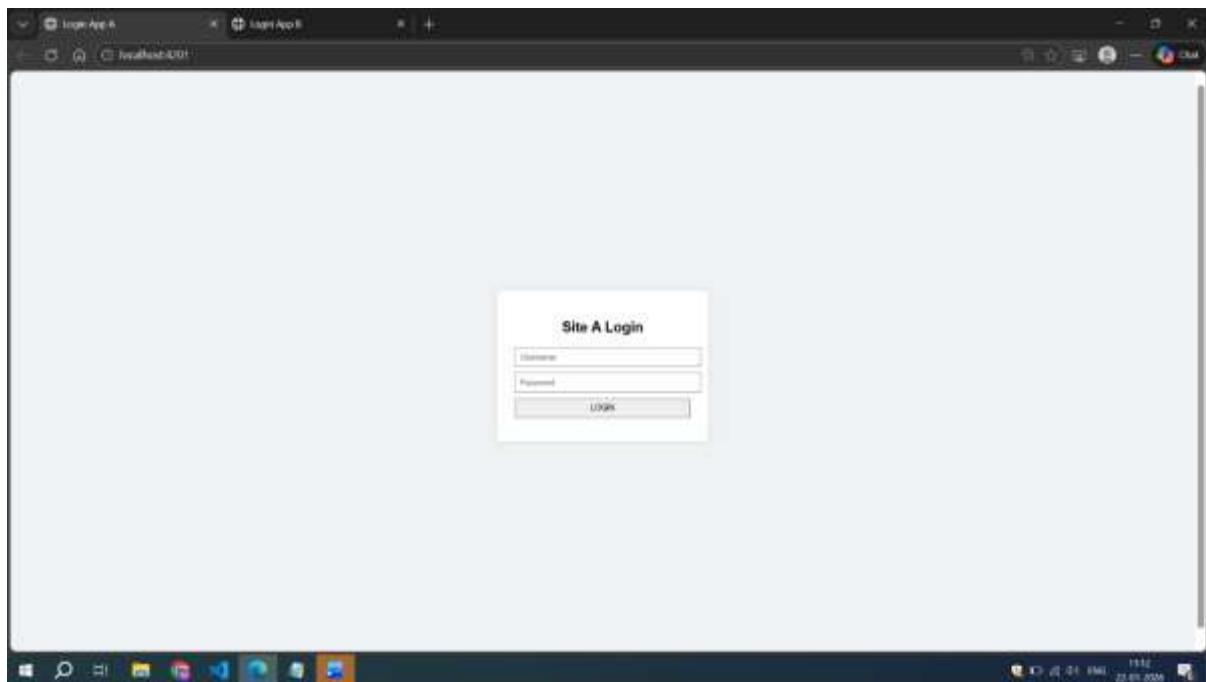


The screenshot shows a code editor with the file `index.html` open. The code is an HTML form for logging in, similar to SiteA but with different input IDs:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Login App</title>
  <style>
    </style>
</head>
<body>
  <form id="loginForm" onsubmit="login(event)">
    <div>A login form</div>
    <input id="user-name" placeholder="username" />
    <input id="pass-word" type="password" />
    <button id="login-button" type="submit">Login</button>
  </form>
  <script>
    function login(e) {
      e.preventDefault();
      window.location.href = "dashboard.html";
    }
  </script>
</body>
</html>
```

Below the code, the terminal shows:

```
PS C:\Users\vinay\OneDrive\Projects\SiteB\siteB> node .\server.js
PS C:\Users\vinay\OneDrive\Projects\SiteB\siteB> node .\server.js
Site B is running at http://localhost:4000
```



2 test files

The screenshot shows a browser window with developer tools open. The left sidebar displays a file tree under 'OPEN EDITORS' and 'GLOBAL PROJECT'. The main area shows a code editor with a file named 'login.spec.js'. The code is a Mocha test suite for a login feature, using Chai assertions and a custom 'logintage' helper function. The bottom status bar shows several tabs and the path 'File:///C:/Users/.../Desktop/test-project'. The bottom right corner has a small preview of the browser window.

```
import { test } from "playwright/test";
import * as dotenv from "dotenv";
import { Logintage } from "./pages/loginPage.js";
import { Logindata } from "./data/logindata.js";

dotenv.config();
const SITEURL = process.env.SITE_H_URL;

test.describe("Login tests", () => {
  test.beforeEach(async ({ page }) => {
    const logintage = new Logintage(page);
    await logintage.open(xPath);
    page.goto(SITEURL + logintage);
  });

  test("Valid login", async ({ page }) => {
    const logintage = page.logintage;

    await logintage.login(logindata.username, logindata.password);
    await logintage.verifyHome();
  });
});

//
```

The screenshot shows a browser window with developer tools open. The left sidebar displays a file tree for a project named 'CLASSE-PROJECT'. The 'src' folder contains files like 'index.html', 'index.css', 'index.js', 'index.spec.js', 'login.js', 'login.spec.js', 'login.html', 'login.css', 'login.selector.js', 'login.selector.spec.js', 'login.page.js', and 'login.page.spec.js'. The 'login.page.spec.js' file is currently selected and shown in the main code editor area. The code in the editor is a Jest test for a login page, using the Page Object Model (POM) pattern. It includes imports for 'LoginPage', 'Page', and 'expect'. The test cases involve opening the login page, entering credentials ('username' and 'password'), and verifying the dashboard title. The bottom status bar shows the path as 'D:\App\fleximind\CLASSE-PROJECT\src\tests\unit\tests' and the file name as 'login.page.spec.js'. The bottom right corner shows the browser's address bar with 'localhost:3001'.

```
export class LoginPage {
  constructor(page) {
    this.page = page;
  }

  this.usernameField = page.locator(loginSelectors.usernameField);
  this.passwordField = page.locator(loginSelectors.passwordField);
  this.loginButton = pagelocator(loginSelectors.loginButton);
  this.dashboardTitle = page.locator(loginSelectors.dashboardTitle);
}

async open(url) {
  await this.page.goto(url);
}

async login(username, password) {
  await this.usernameField.fill(username);
  await this.passwordField.fill(password);
  await this.loginButton.click();
}

async verifyDashboard() {
  await expect(this.page).toHaveTitle('Dashboard');
  await expect(this.dashboardTitle).toContainText('Dashboard');
}

it('should log in successfully', async () => {
  await login('admin', 'admin');
  await verifyDashboard();
})
```

The screenshot shows a Windows desktop environment with a terminal window open in the foreground. The terminal window has tabs for 'login.html', 'logindata.js', and 'logindata.html'. The current tab, 'logindata.js', contains the following code:

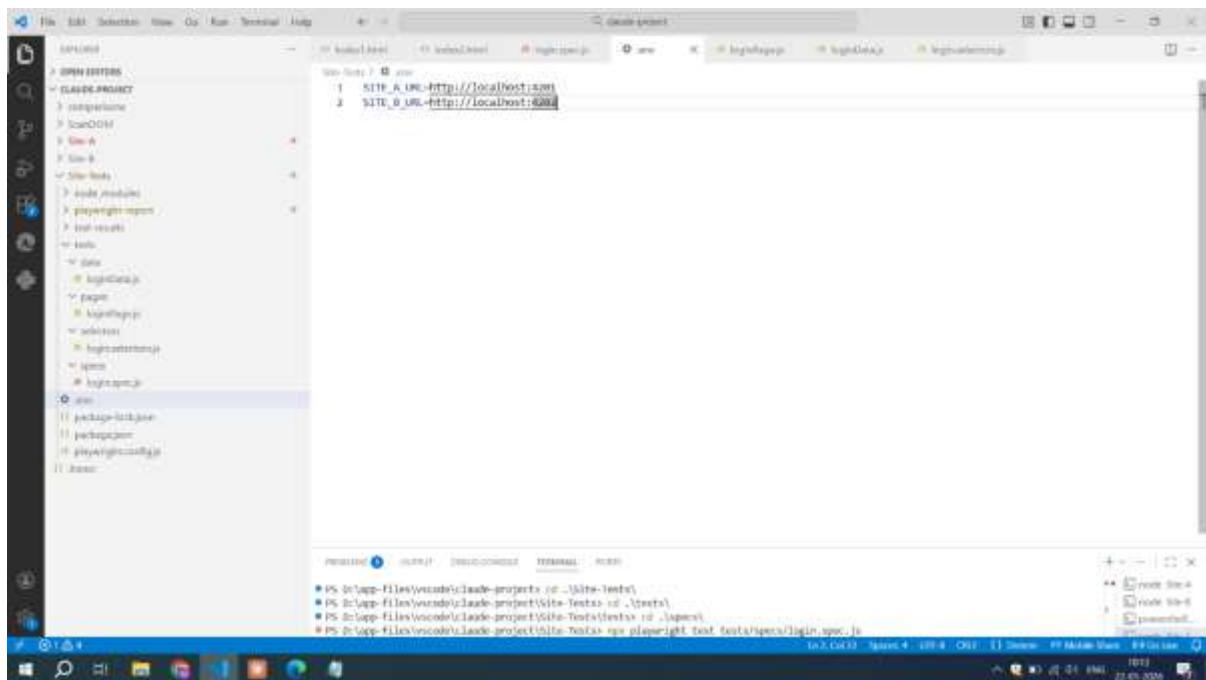
```
const logindata = [
  {
    username: "admin",
    password: "password"
  }
]
```

The terminal below the window shows the command PS > inApp-Files\vscode\claude-projects\site-test\src\tests, indicating the current working directory.

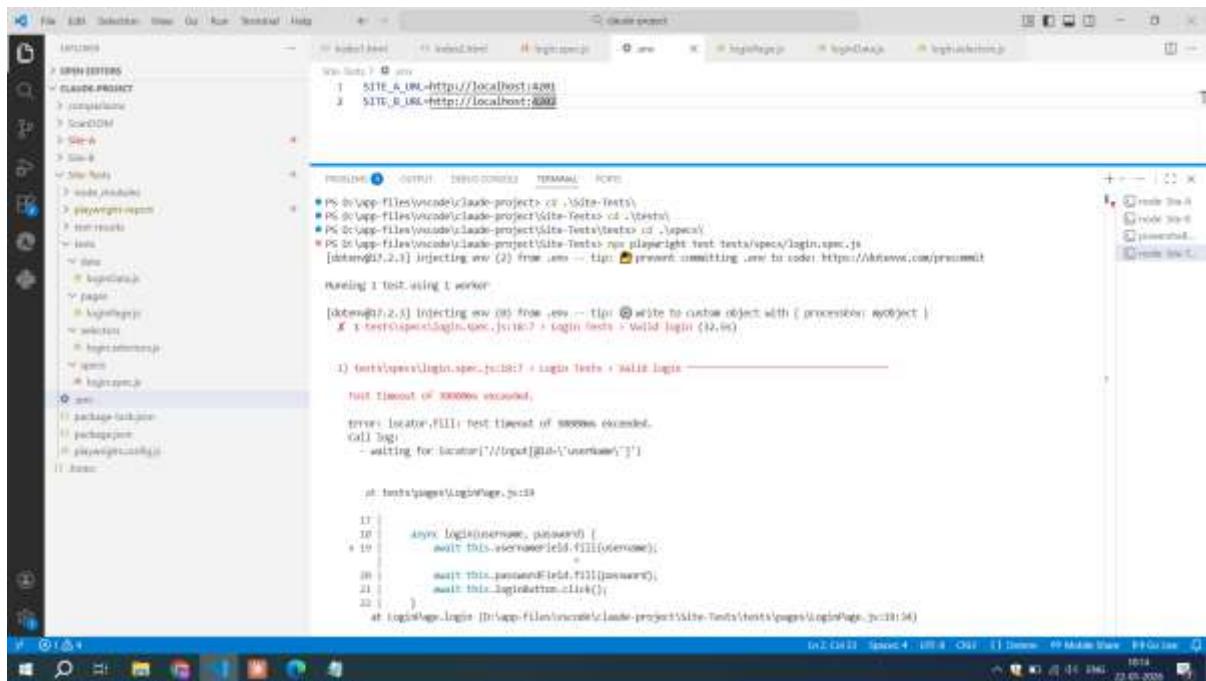
The screenshot shows a Windows desktop environment with a terminal window open in the foreground. The terminal window has tabs for 'login.html', 'logindata.js', and 'logindata.html'. The current tab, 'logindata.js', contains the following code:

```
export const logindata = [
  {
    usernameId: "//input[@id='username']",
    passwordId: "//input[@id='password']",
    loginButton: "//input[@type='submit' and @value='Login']"
  }
]
```

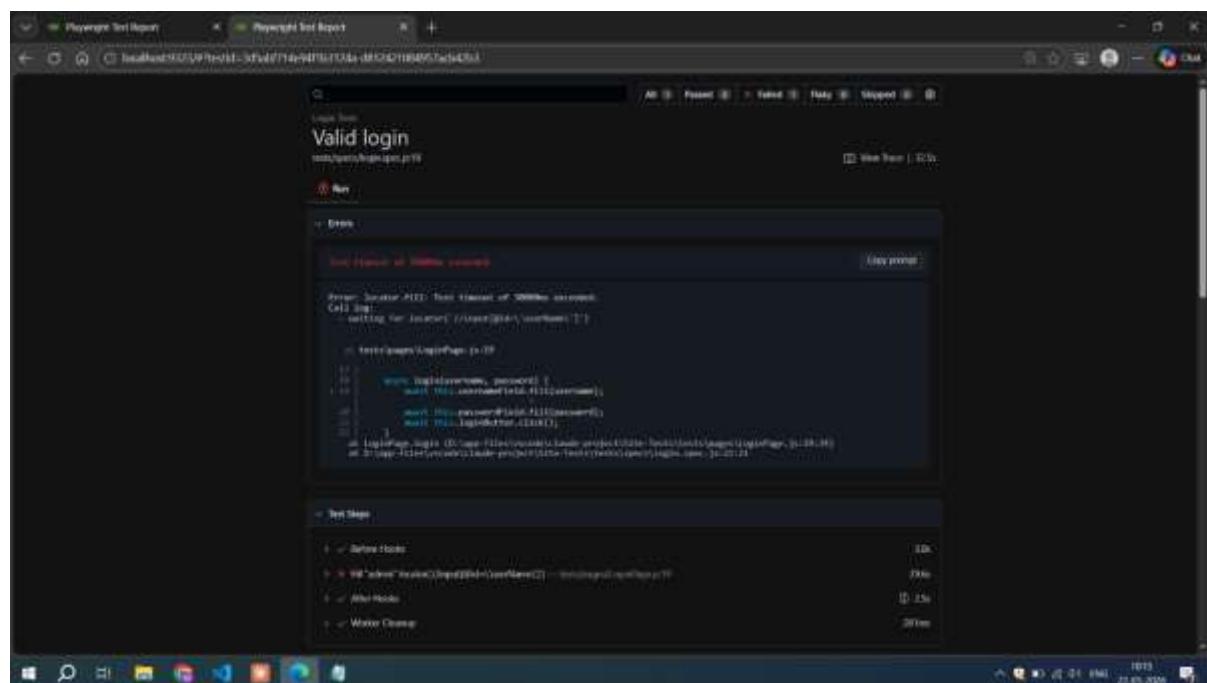
The terminal below the window shows the command PS > inApp-Files\vscode\claude-projects\site-test\src\tests, indicating the current working directory.



3) test result in terminal (npx playwright test)

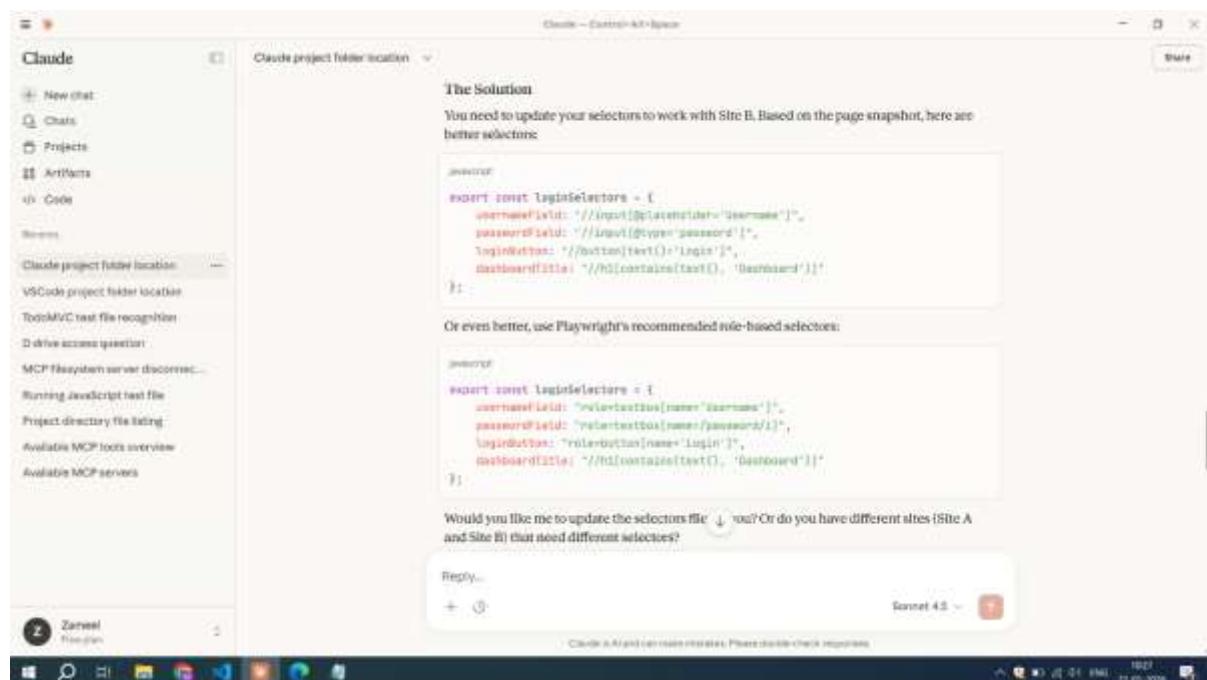


4) playwright report



5) agentic ai interface(Claude)

i) analysing the error



Claude

+ New chat
Q Chats
D Projects
II Artifacts
O Code

Recent
Claude project folder location
VSCode project folder location
TodoMVC test file recognition
D drive access question
MCP filesystem server disconnect
Running JavaScript test file
Project directory file listing
Available MCP tools overview
Available MCP servers

Claude project folder location

do you know this folder
D:\app-files\vscode\claude-project
??

I don't have direct access to your local file system, so I can't see what's in that folder on your D: drive. However, I can help you explore it if you'd like!

If you want me to see what's in that directory, you could:

1. Share the folder contents with me - you can copy and paste the file/folder structure here
2. If you're using the Claude desktop app with computer use permissions, I might be able to help navigate it.
3. Upload specific files from that folder that you'd like me to look at or work with

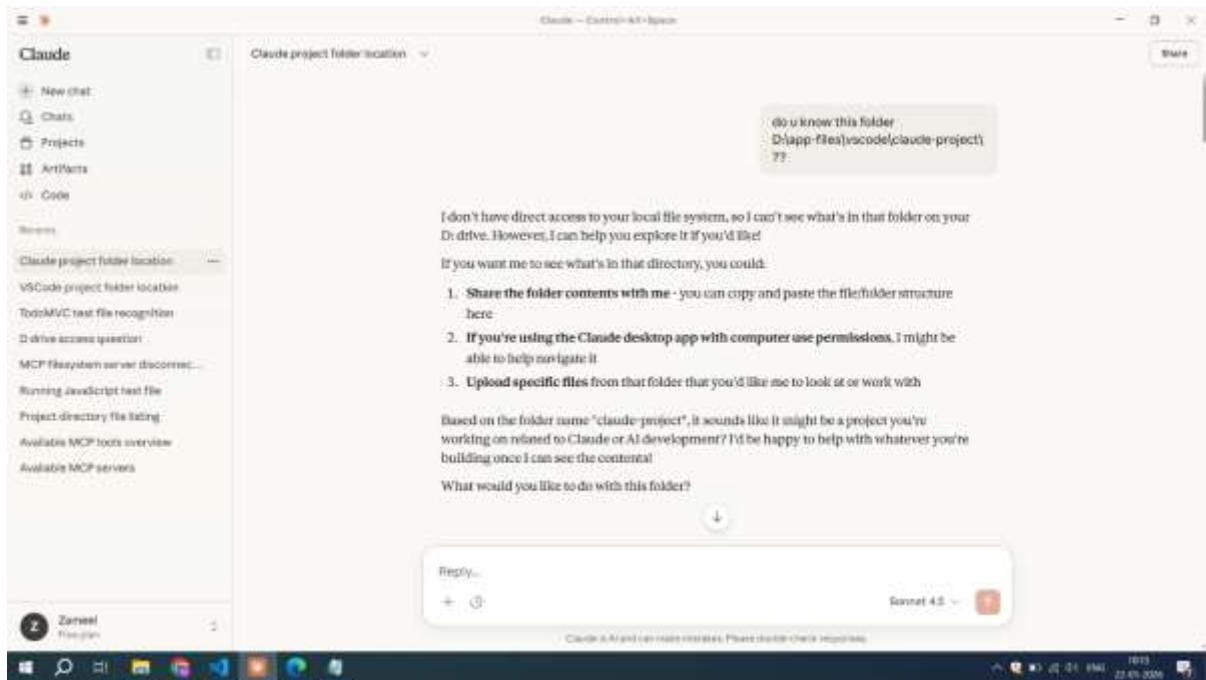
Based on the folder name "claude-project", it sounds like it might be a project you're working on related to Claude or AI development? I'd be happy to help with whatever you're building once I can see the contents!

What would you like to do with this folder?

Reply...
+ ⚡

Gานet 4.5

Claude AI and API usage exceeded. Please check claude_usage.info



Claude

+ New chat
Q Chats
D Projects
II Artifacts
O Code

Recent
Claude project folder location
VSCode project folder location
TodoMVC test file recognition
D drive access question
MCP filesystem server disconnect
Running JavaScript test file
Project directory file listing
Available MCP tools overview
Available MCP servers

Claude project folder location

What would you like to do with this folder?

do you have filesystem protocol right ?

You're absolutely right! I do have filesystem access capabilities. Let me check if I can access that directory.

r: /etc/allowed_directories

Great! I do have access to the D: drive. Let me check out that folder for you.

r: /etc_directory

Yes I can see your `claude-project` folder. It contains:

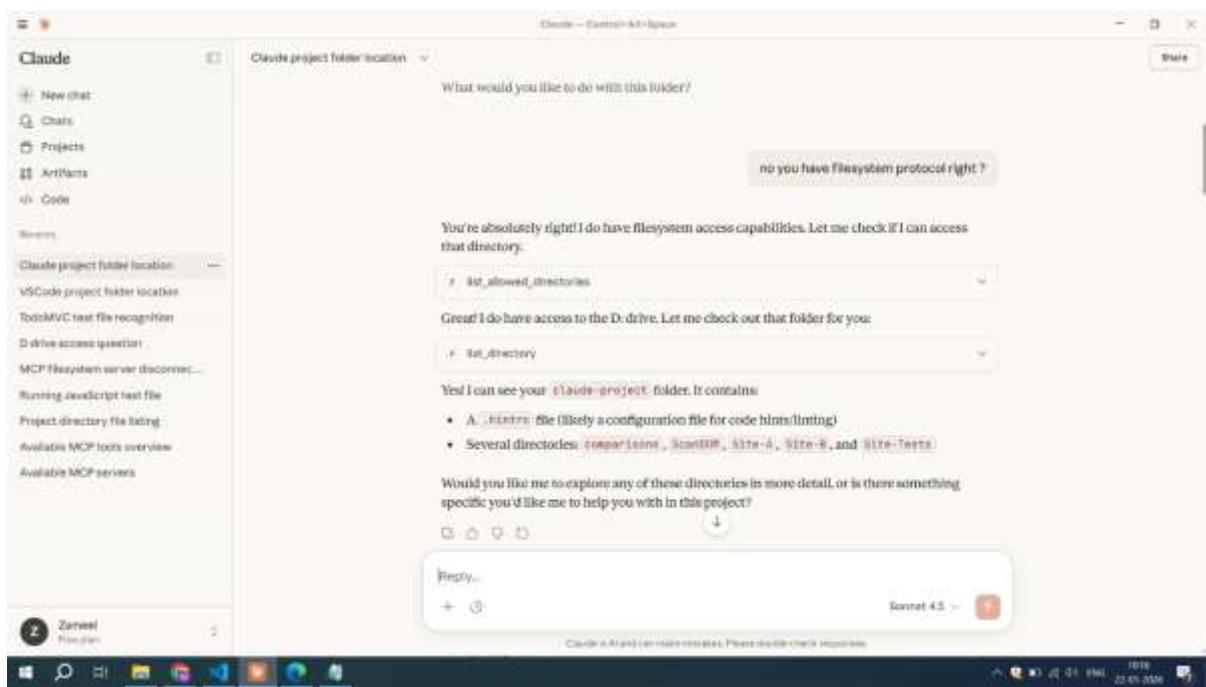
- A `.eslintrc` file (likely a configuration file for code hints/linting)
- Several directories: `comparison`, `static`, `site-A`, `site-B`, and `site-Tests`

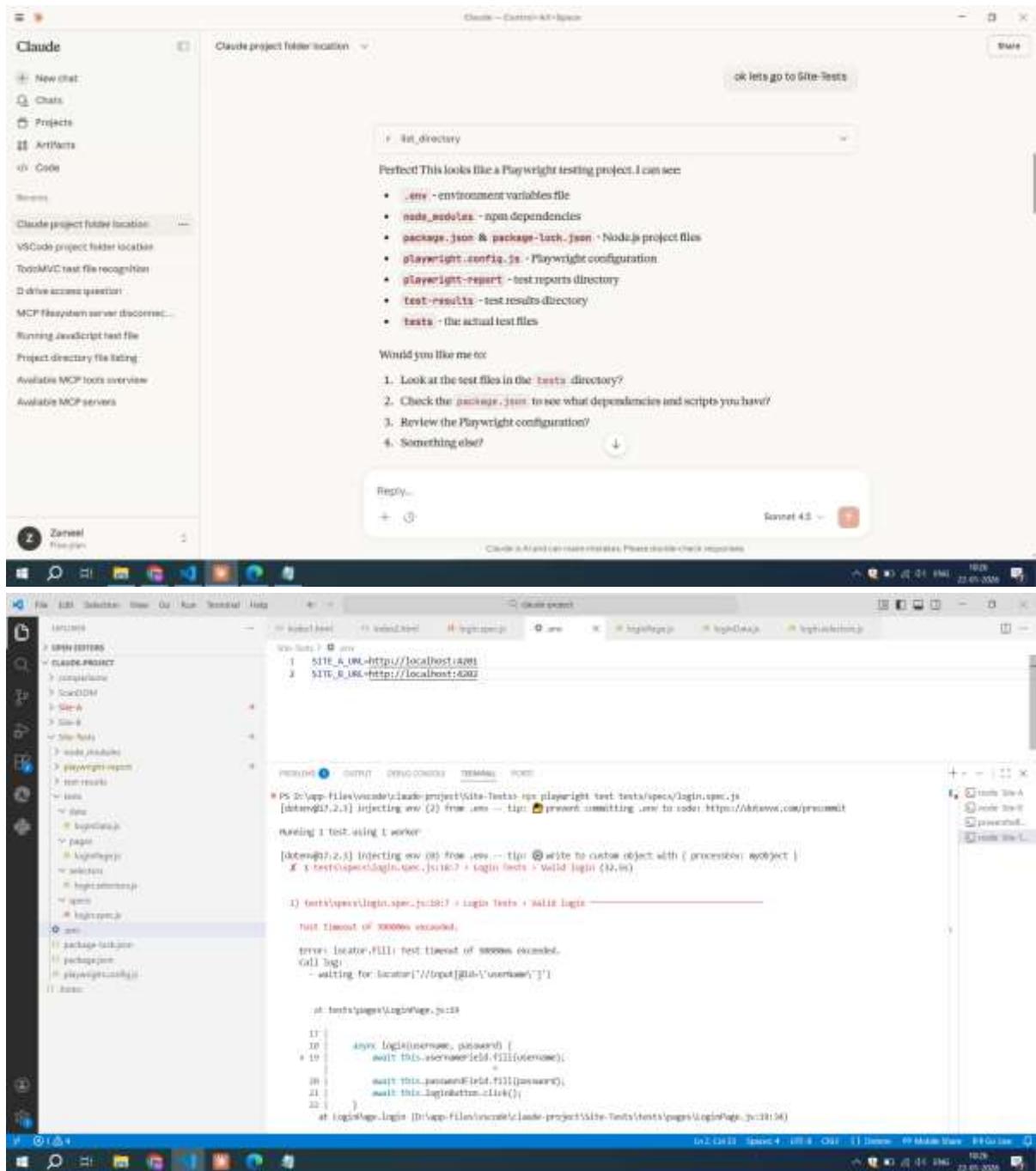
Would you like me to explore any of these directories in more detail, or is there something specific you'd like me to help you with in this project?

Reply...
+ ⚡

Gานet 4.5

Claude AI and API usage exceeded. Please check claude_usage.info





Claude

New chat Chats Projects Artifacts Code

Claude project folder location

VSCode project folder location

TodoMVC test file recognition

D drive access question

MCP filesystem server disconnect

Running JavaScript test file

Project directory file listing

Available MCP icons overview

Available MCP servers

I can see the test is timing out while trying to find the username field with XPath
`//input[@id='username']`. Let me investigate this issue by looking at your test files and the screenshot/error context.

10:00 AM

read_text_file

read_text_file

Now let me check the error context and screenshot to see what's happening:

read_text_file

Perfect! I found the issue. Looking at the page snapshot, the actual page structure doesn't match your selectors. The page has:

- A textbox with accessible name "Username"
- Another textbox (likely password)
- A button "Login"

But your selectors are looking for `input[@id='username']` and `input[@id='password']` which don't exist on the page.

Let me check the screenshot to confirm:

Reply...

Screenshot 4.5

Zaneel

10:00 AM 22.05.2024

Claude

New chat Chats Projects Artifacts Code

Claude project folder location

VSCode project folder location

TodoMVC test file recognition

D drive access question

MCP filesystem server disconnect

Running JavaScript test file

Project directory file listing

Available MCP icons overview

Available MCP servers

Let me check the screenshot to confirm:

read_text_file

Perfect! I can see the login page. The issue is clear: your selectors are incorrect for Site B. The page has a simple form with placeholder text "Username" but your XPath selectors are looking for specific IDs that don't exist.

The Problem

Your selectors in `LoginSelectors.js` are:

- `//input[@id='username']`
- `//input[@id='password']`
- `//input[@type='submit' and @value='LOGIN']`

But Site B's login page doesn't have these IDs or a submit input - it has a button instead.

The Solution

You need to update your selectors to work with Site B. Based on the page snapshot, here are better selectors:

```
import { Selector } from 'testcafe'

export const loginSelectors = {
    username: Selector('#username'),
    password: Selector('#password'),
    loginButton: Selector('button[value="LOGIN"]')
}
```

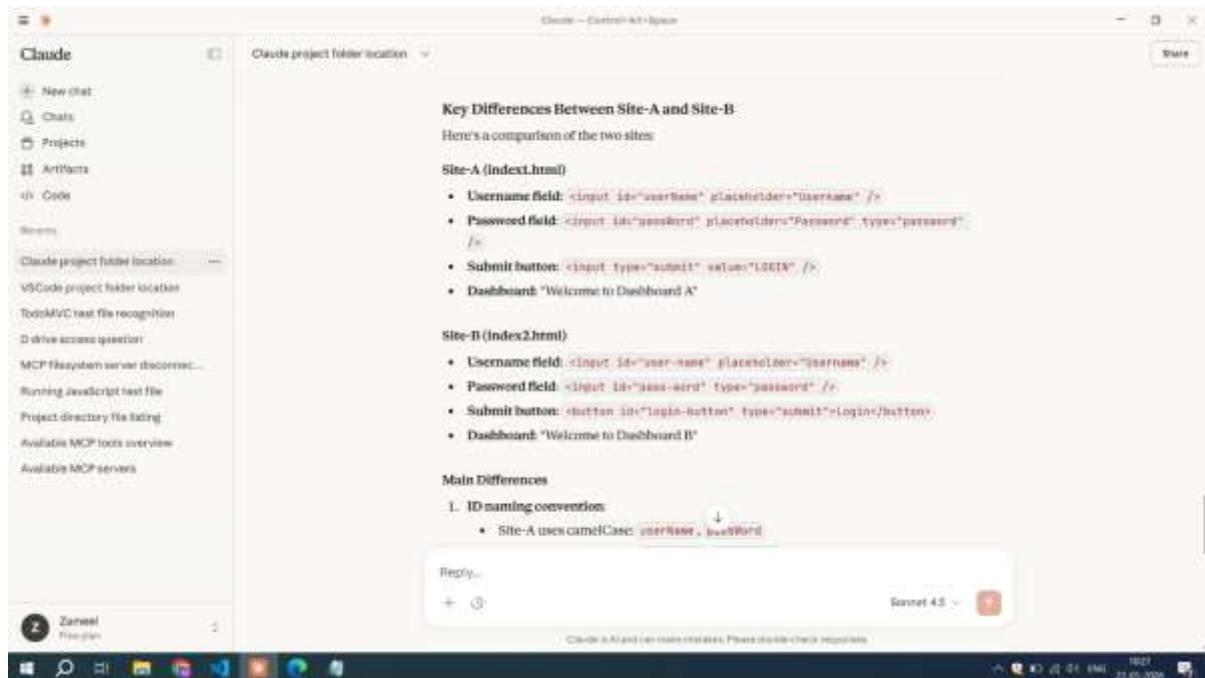
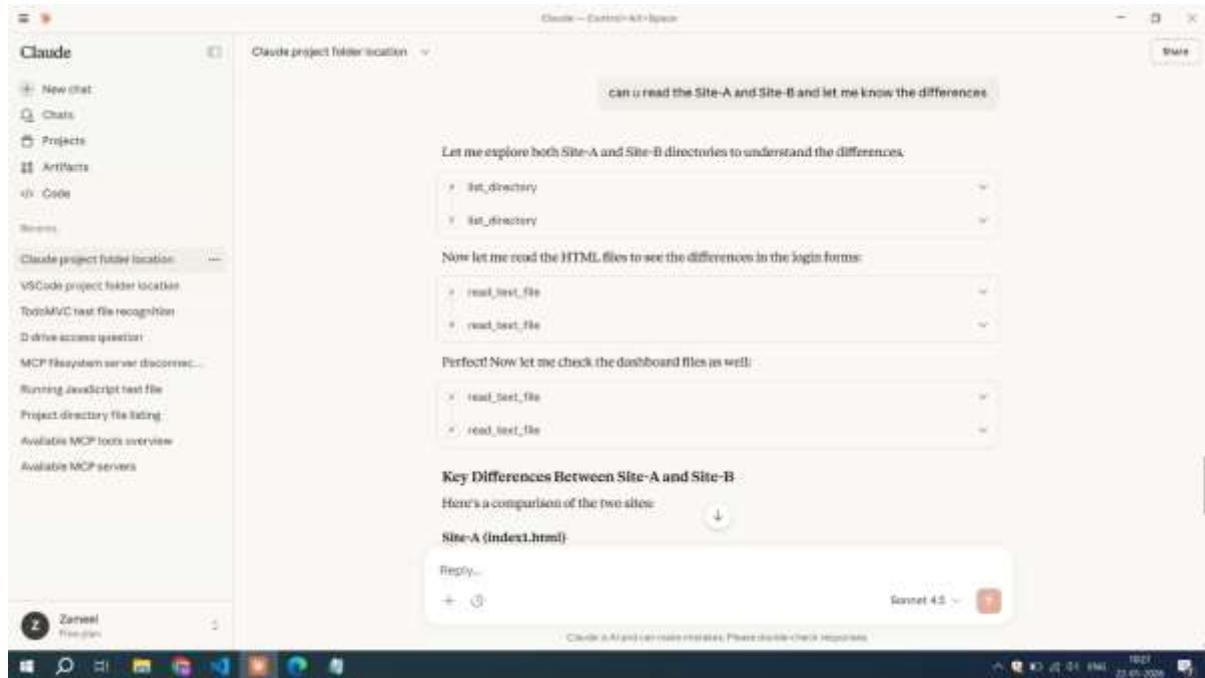
Reply...

Screenshot 4.5

Zaneel

10:00 AM 22.05.2024

ii)using the files of the Sites used



Claude – Claude-AI-Space

Claude project folder location

Main Differences

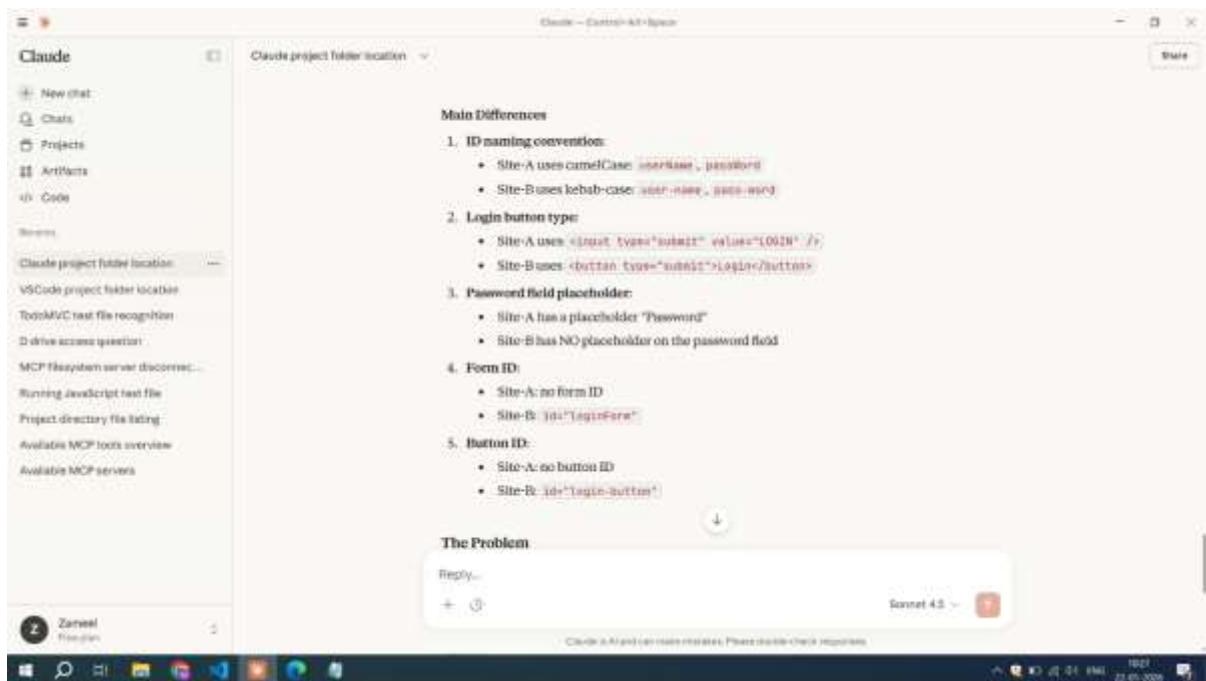
1. ID naming convention
 - Site-A uses camelCase: `username`, `password`
 - Site-B uses kebab-case: `user-name`, `pass-word`
2. Login button type:
 - Site-A uses: `<input type="submit" value="LOGIN" />`
 - Site-B uses: `<button type="submit">Login</button>`
3. Password field placeholder:
 - Site-A has a placeholder "Password"
 - Site-B has NO placeholder on the password field
4. Form ID:
 - Site-A: no form ID
 - Site-B: `id="loginForm"`
5. Button ID:
 - Site-A: no button ID
 - Site-B: `id="login-button"`

The Problem

Reply...
+ ⌂

Syntax 4.5. ✓

Claude AI and GPT have interacted. Please double-check dependencies.



Claude – Claude-AI-Space

Claude project folder location

The Problem

Your current selectors are designed for **Site-A** but you're testing against **Site-B**. That's why the test is failing!

Current selectors (match Site-A):

```
usernameField: "//input[@id='username']",
passwordField: "//input[@id='password']",
loginButton: "//input[@type='submit' and @value='LOGIN']";
```

What Site-B needs:

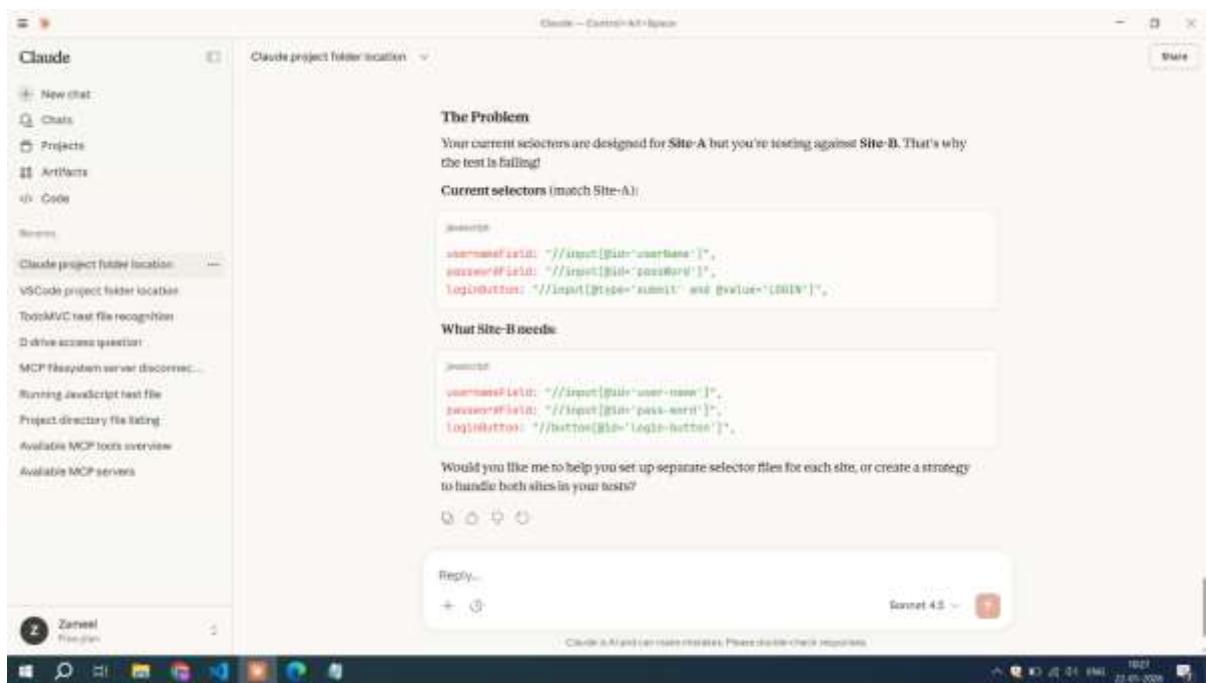
```
usernameField: "//input[@id='user-name']",
passwordField: "//input[@id='pass-word']",
loginButton: "//button[@id='login-button']";
```

Would you like me to help you set up separate selector files for each site, or create a strategy to handle both sites in your tests?

Reply...
+ ⌂

Syntax 4.5. ✓

Claude AI and GPT have interacted. Please double-check dependencies.



iii) using ScanDOM and its results

The screenshot shows a Windows desktop environment with a code editor, terminal, and file browser.

Code Editor: The main window displays a file named `baseline.js` with the following content:

```
const fs = require('fs');
const path = require('path');
const http = require('http');
const https = require('https');
const cheerio = require('cheerio');
const puppeteer = require('puppeteer');
const { JSDOM } = require('jsdom');

const SITE_A_URL = 'https://www.google.com';
const SITE_B_URL = 'https://www.bing.com';

if (!SITE_A_URL || !SITE_B_URL) {
  console.error(`Please set SITE_A_URL and SITE_B_URL in your .env file`);
  process.exit(1);
}

async () => {
  console.log(`Starting baseline & variant scanning...`);

  await scanSite(SITE_A_URL);
  await scanSite(SITE_B_URL);

  console.log(`Completed scanning both sites.`);
}();

function scanSite(url) {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto(url);
  const content = await page.content();
  const dom = new JSDOM(content);
  const $ = cheerio.load(dom.window.document);
  const title = $('title').text();
  const hrefs = $('a').attr('href');
  const imgSrcs = $('img').attr('src');
  const scriptSrcs = $('script').attr('src');
  const styleSrcs = $('style').attr('src');

  const screenshots = [
    ...imgSrcs.map(src => `Screenshot + ${src}`),
    ...scriptSrcs.map(src => `Screenshot + ${src}`),
    ...styleSrcs.map(src => `Screenshot + ${src}`),
  ];

  const links = hrefs.map(href => `Link: ${href}`);
  const titles = [title];
  const images = imgSrcs;
  const scripts = scriptSrcs;
  const styles = styleSrcs;

  browser.close();
  return { url, titles, links, images, scripts, styles, screenshots };
}

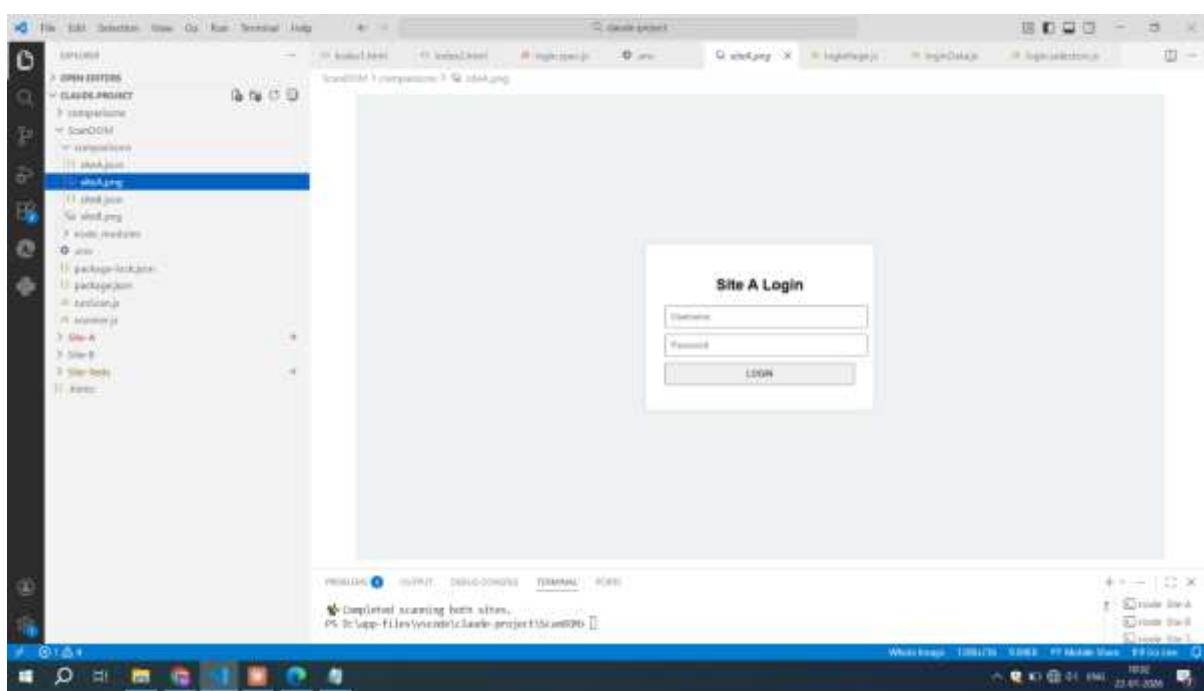
process.env.NODE_ENV === 'production' ? scanSite(SITE_A_URL) : scanSite(SITE_B_URL);
```

Terminal: The bottom window shows the command line output:

```
% cd app-Files/scalene/clade-project/scalene
% node ./scalene.js
[info]:@12.1.1] Injecting site (1) from .env -- tip: ⚡ write to static object with { processEnv: myObject }
⚡ Starting baseline & variant scanning...
⚡ Scanning: http://localhost:4200
⚡ Saved 300M + ./comparison/sites.html
⚡ Saved Screenshot + ./comparison/sites.png
⚡ Scanning: http://localhost:4200
⚡ Saved 700B + ./comparison/sites.json
⚡ Saved Screenshot + ./comparison/sites.png
⚡ Completed scanning both sites.
%
```

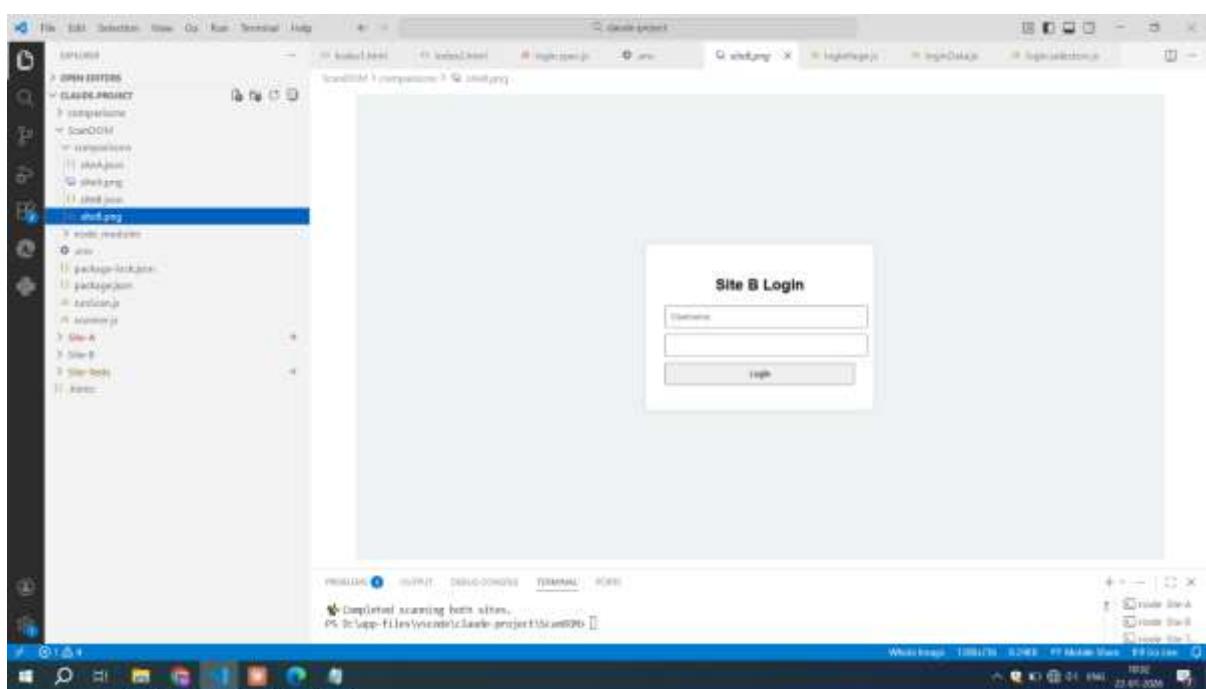
The screenshot shows a Java IDE interface with a menu bar (File, Edit, Selection, View, Go, Run, Terminal, Help) and a toolbar. The title bar says "client project". The left sidebar shows a package structure with a selected package named "com". The main editor area contains Java code for a class named "Client". The code includes imports for java.util, java.net, and javax.swing, as well as a class definition for Client with various methods like start, stop, and handleInput.

```
1 package com;
2
3 import java.util.*;
4 import java.net.*;
5 import javax.swing.*;
6
7 public class Client {
8     // Constructor, methods, and other code
9 }
```



```
1  "scannedAt": "2020-01-22T05:06:38.943Z",
2  "importUri": "http://localhost:4809",
3  "finalUri": "http://localhost:4809",
4  "title": "Login app B",
5  "html": "<!DOCTYPE html><html lang=\"en\"><head><meta charset='UTF-8'><meta name='viewport' content='width=device-width, init-
6  "ializedText": "Site B login page",
7  "buttons": [
8    "Login"
9  ],
10  "inputs": [
11    {
12      "id": "username",
13      "name": "",
14      "type": "text",
15      "placeholder": "username"
16    },
17    {
18      "id": "password",
19      "name": "",
20      "type": "password",
21      "placeholder": ""
22    }
23  ],
24  "lines": []
25 }]
```

The terminal window shows the message: "Completed scanning bems sites, PS: D:\app\FileSystem\oracle\claude-project\SiteB\src\main\java



Client - Control+Alt+Space

Claude

- + New chat
- Q Chats
- Projects
- Artifacts
- Code

Streams

Claude project folder location

VSCode project folder location

TodoMVC test file recognition

D drive access question

MCP filesystem server disconnect

Running JavaScript test file

Project directory file listing

Available MCP tools overview

Available MCP servers

Zameel

Reply...

wait
we have ScanDOM folder right ??
can we use that to find the differences ?

Great idea! Let me check what's in the ScanDOM folder.

+ list_directory

+ list_directory

Perfect! It looks like you already have scan results! Let me check those JSON files.

+ read_text_file

+ read_text_file

Excellent! The ScanDOM tool has already captured the structural differences! Let me also check the scanner code to understand what it does:

+ read_text_file

Perfect! Now I can give you a clear comparison using the ScanDOM data.

↓

Differences Between Site-A and Site-B (from ScanDOM)

Reply...

Share

Gsonet 4.5

Claude is AI and can make mistakes. Please double-check its responses.

Client - Control+Alt+Space

Zameel

Reply...

Differences Between Site-A and Site-B (from ScanDOM)

Site A (localhost:4201)

Inputs:

```
[{"id": "username", "type": "text", "placeholder": "Username"}, {"id": "password", "type": "password", "placeholder": "Password"}, {"id": "", "type": "submit", "placeholder": ""}]
```

Buttons: [] (none - uses submit input)

Site B (localhost:4202)

Inputs:

```
[{"id": "user-name", "type": "text", "placeholder": "Username"}, {"id": "pass-word", "type": "password", "placeholder": ""}]
```

Buttons: ["Login"]

Reply...

Share

Gsonet 4.5

Claude is AI and can make mistakes. Please double-check its responses.

Client - Control+Alt+Space

Zameel

Reply...

Differences Between Site-A and Site-B (from ScanDOM)

Site A (localhost:4201)

Inputs:

```
[{"id": "username", "type": "text", "placeholder": "Username"}, {"id": "password", "type": "password", "placeholder": "Password"}, {"id": "", "type": "submit", "placeholder": ""}]
```

Buttons: [] (none - uses submit input)

Site B (localhost:4202)

Inputs:

```
[{"id": "user-name", "type": "text", "placeholder": "Username"}, {"id": "pass-word", "type": "password", "placeholder": ""}]
```

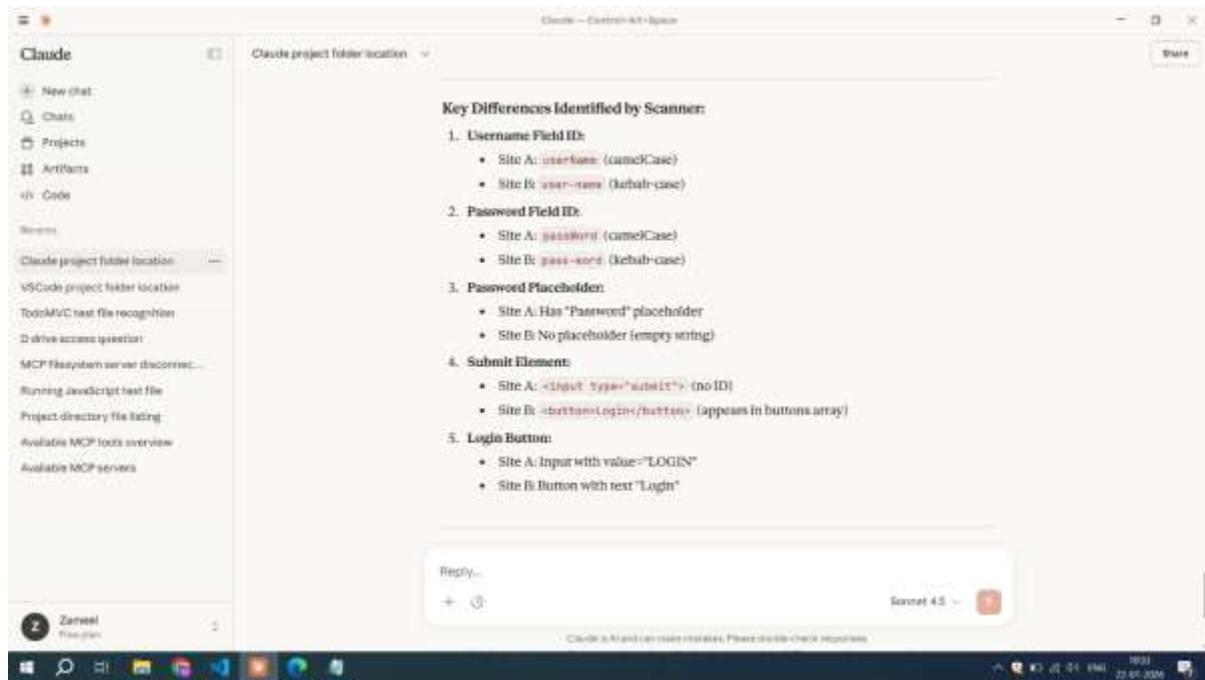
Buttons: ["Login"]

Reply...

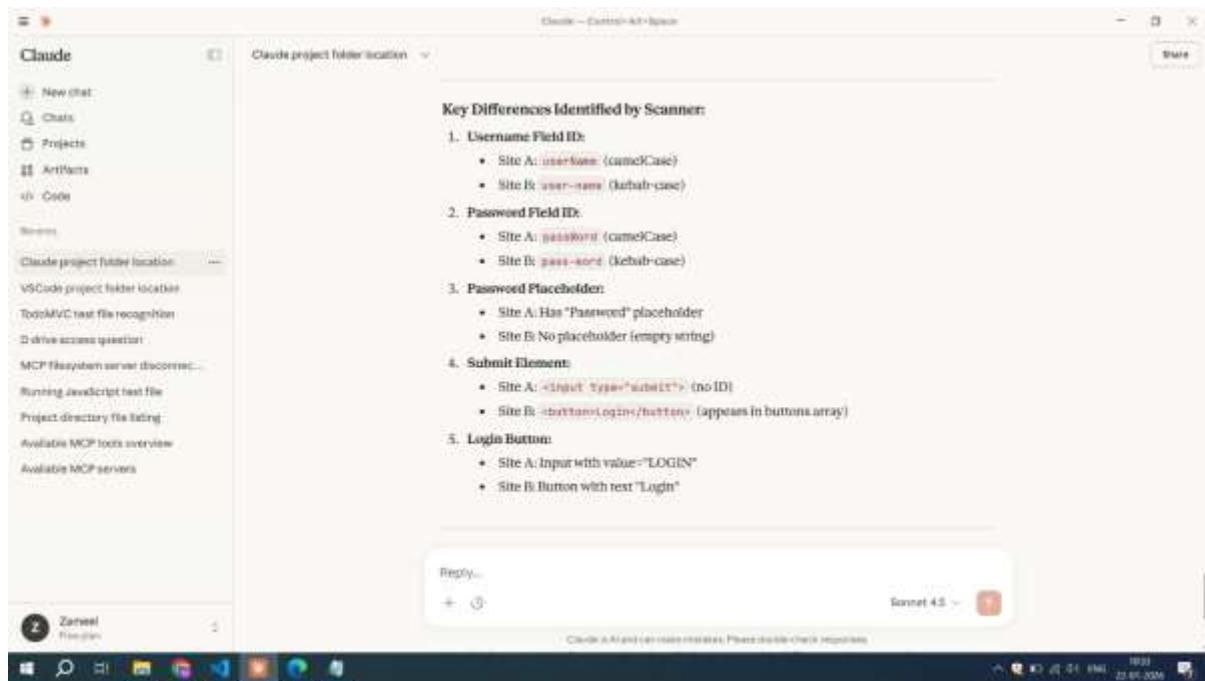
Share

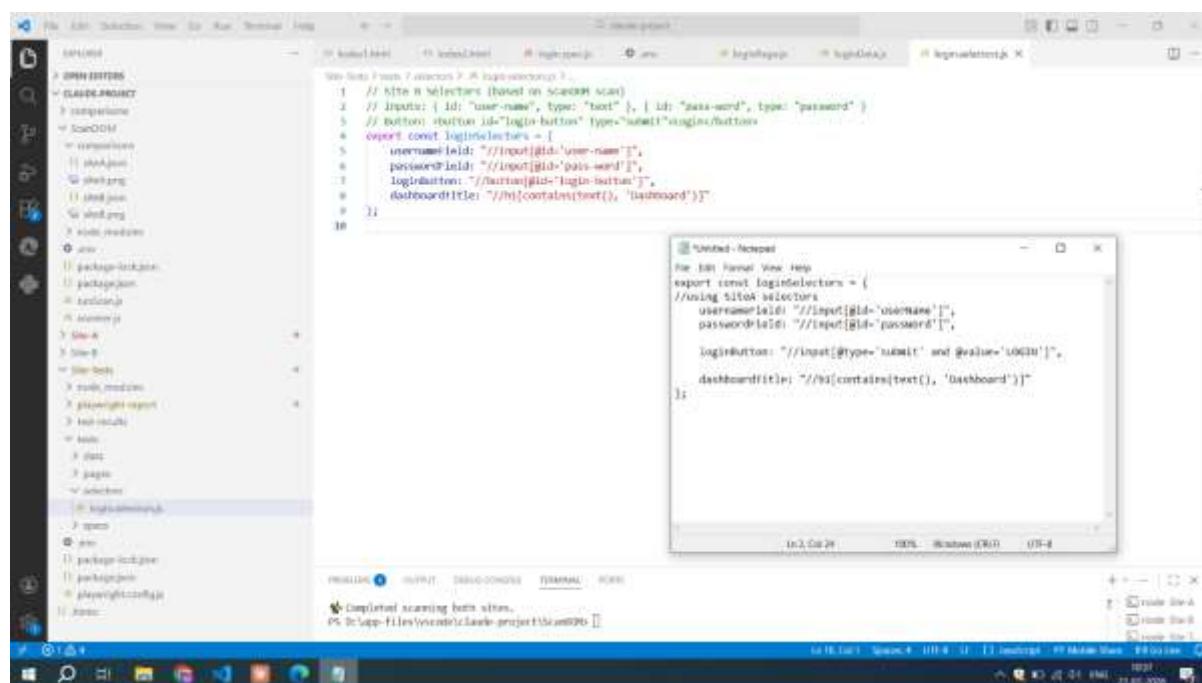
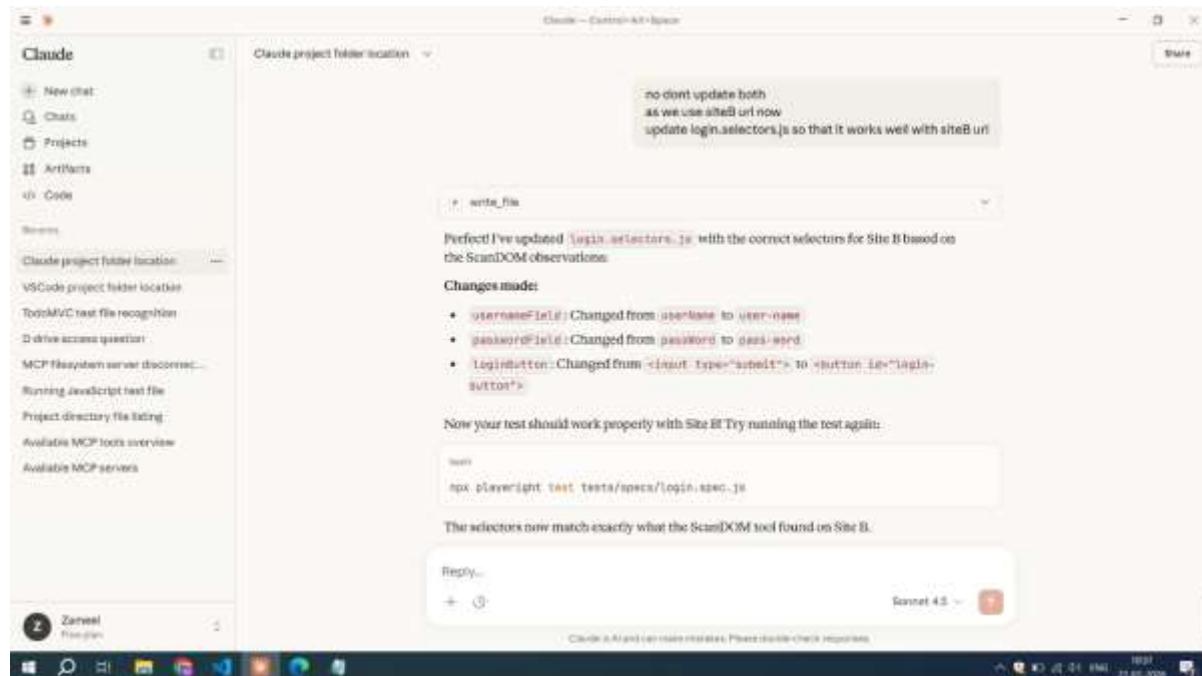
Gsonet 4.5

Claude is AI and can make mistakes. Please double-check its responses.



iv) using Filesystem protocol and Playwright protocol to make changes





6) Running the tests

