**Week 10. Install and Explore Selenium for Automation Testing**

**Step 1: Install Node.js and npm**

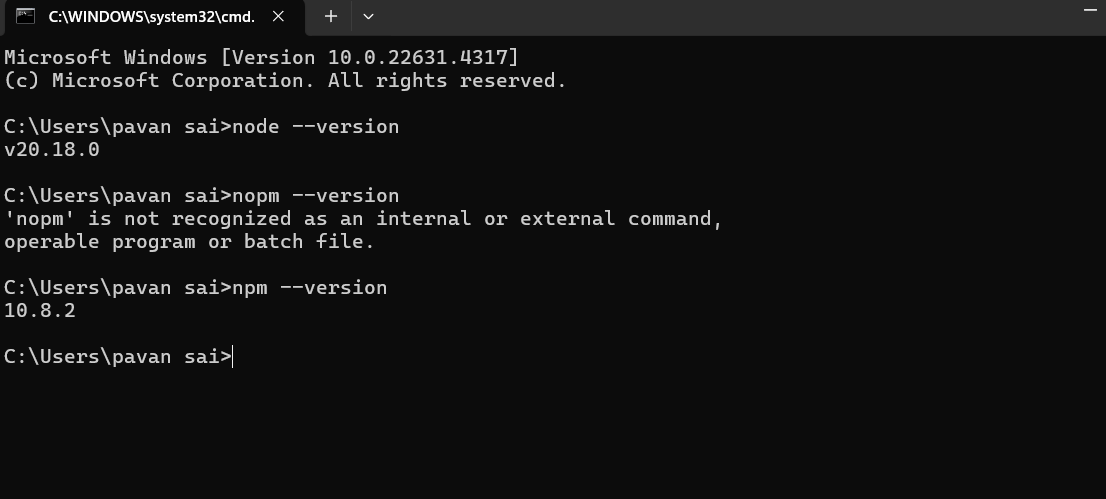
Selenium WebDriver works with JavaScript via Node.js.

1. Download and install Node.js from the [official website](https://nodejs.org).

2. Verify the installation by running:

**node --version**

**npm –version**

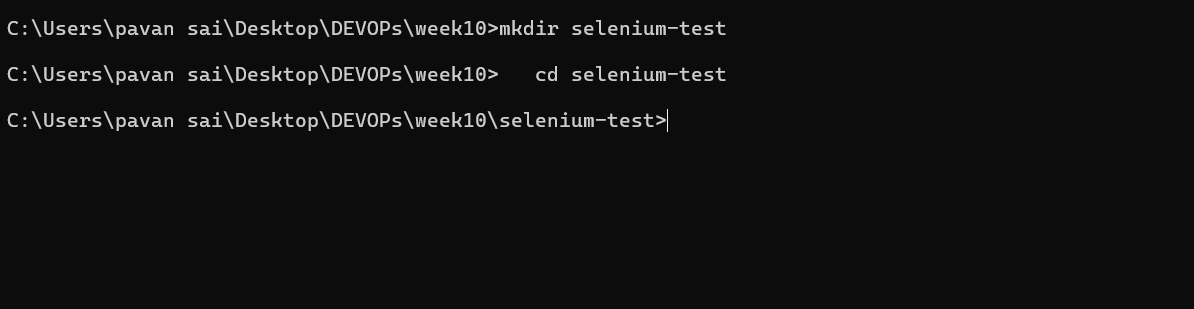
****

**Step 2: Install Selenium WebDriver**

1. Create a new folder for your project:

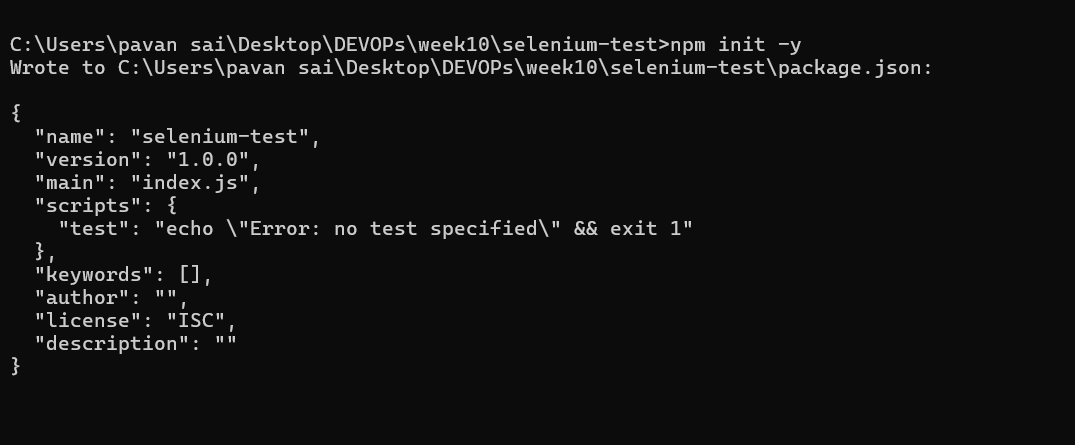
**mkdir selenium-test**

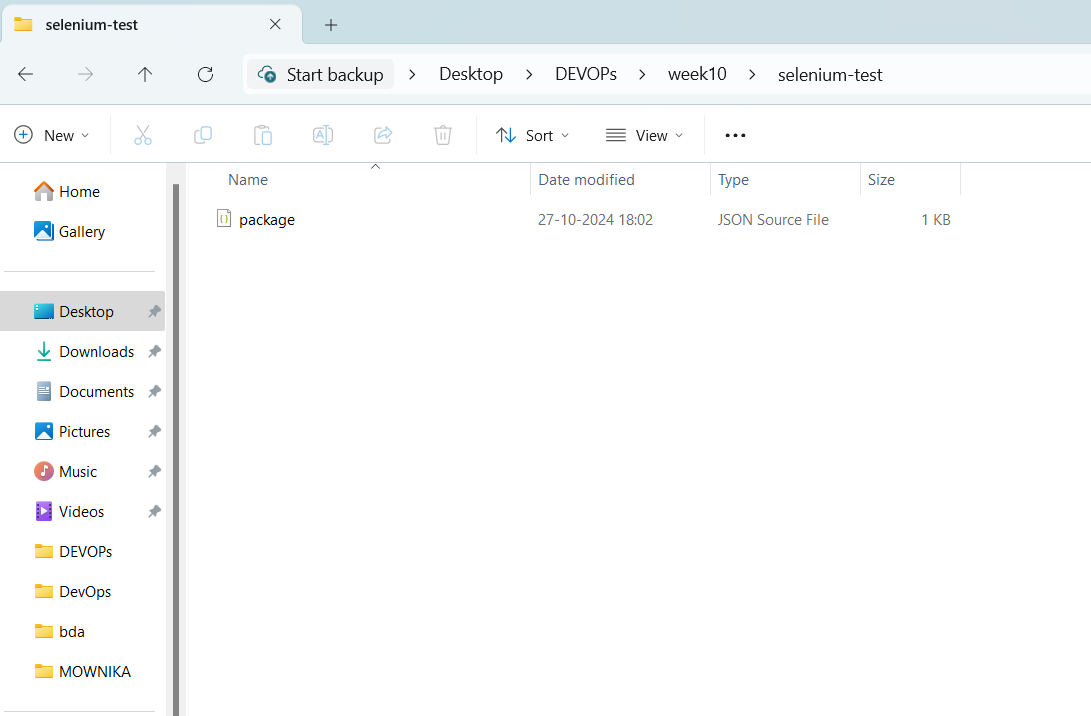
**cd selenium-test**



2. Initialize your Node.js project:

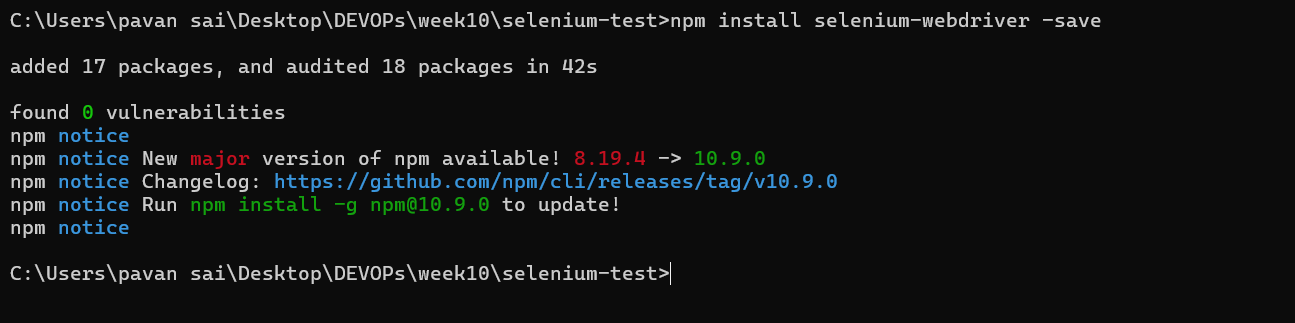
**npm init -y**

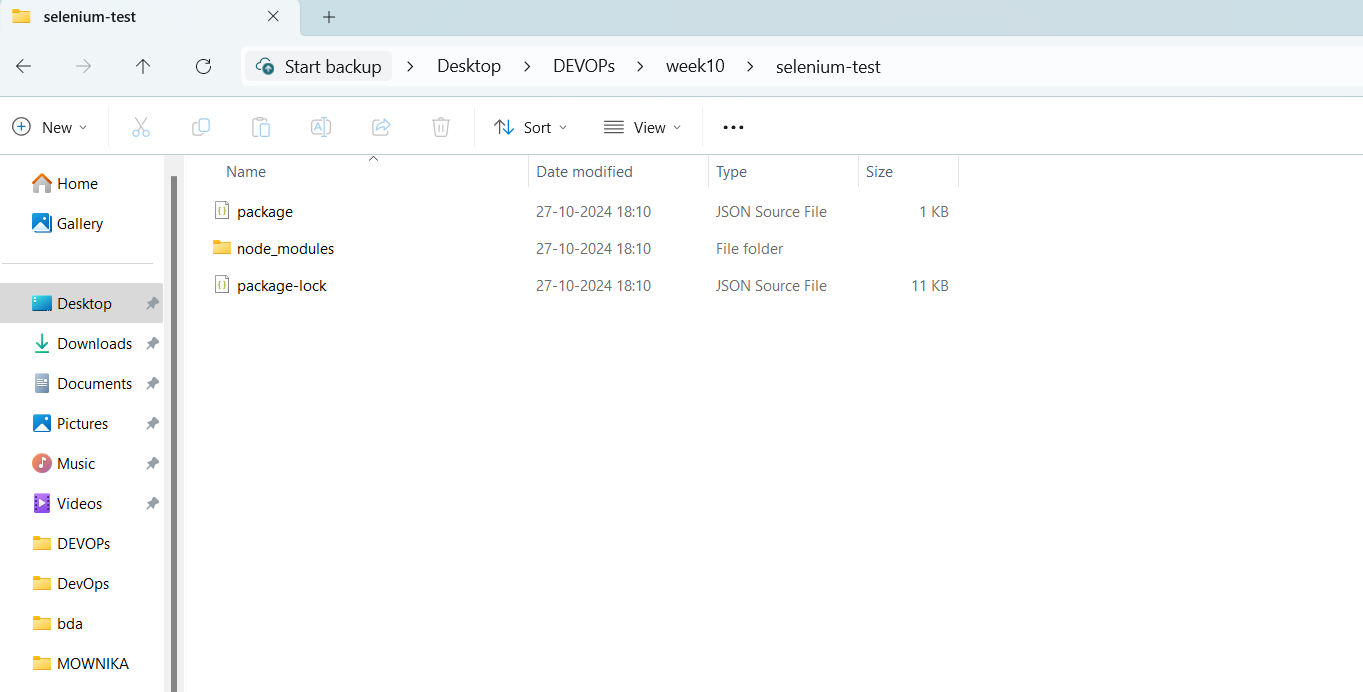
****



**3. Install Selenium WebDriver:**

npm install selenium-webdriver –save

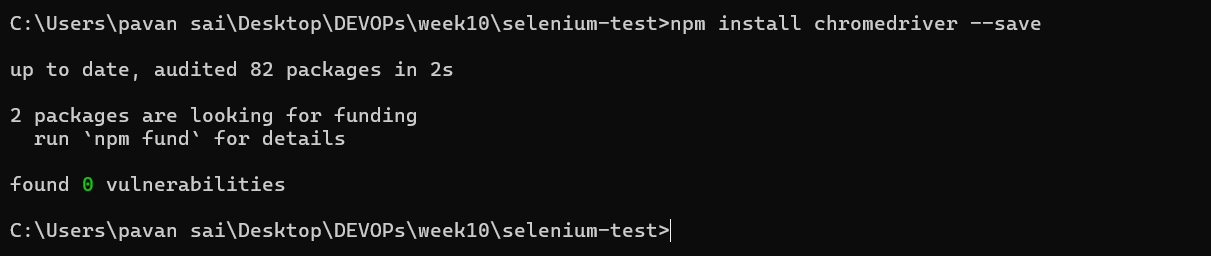




**Step 3: Install Browser Driver**

**1. Install `chromedriver` for Chrome:**

npm install chromedriver –save



**Step 4: Explore Selenium Commands**

Basic Selenium commands

- `driver.get(url)`: Opens a URL.

- `findElement(By.locator)`: Locates an element (e.g., by ID, name, XPath).

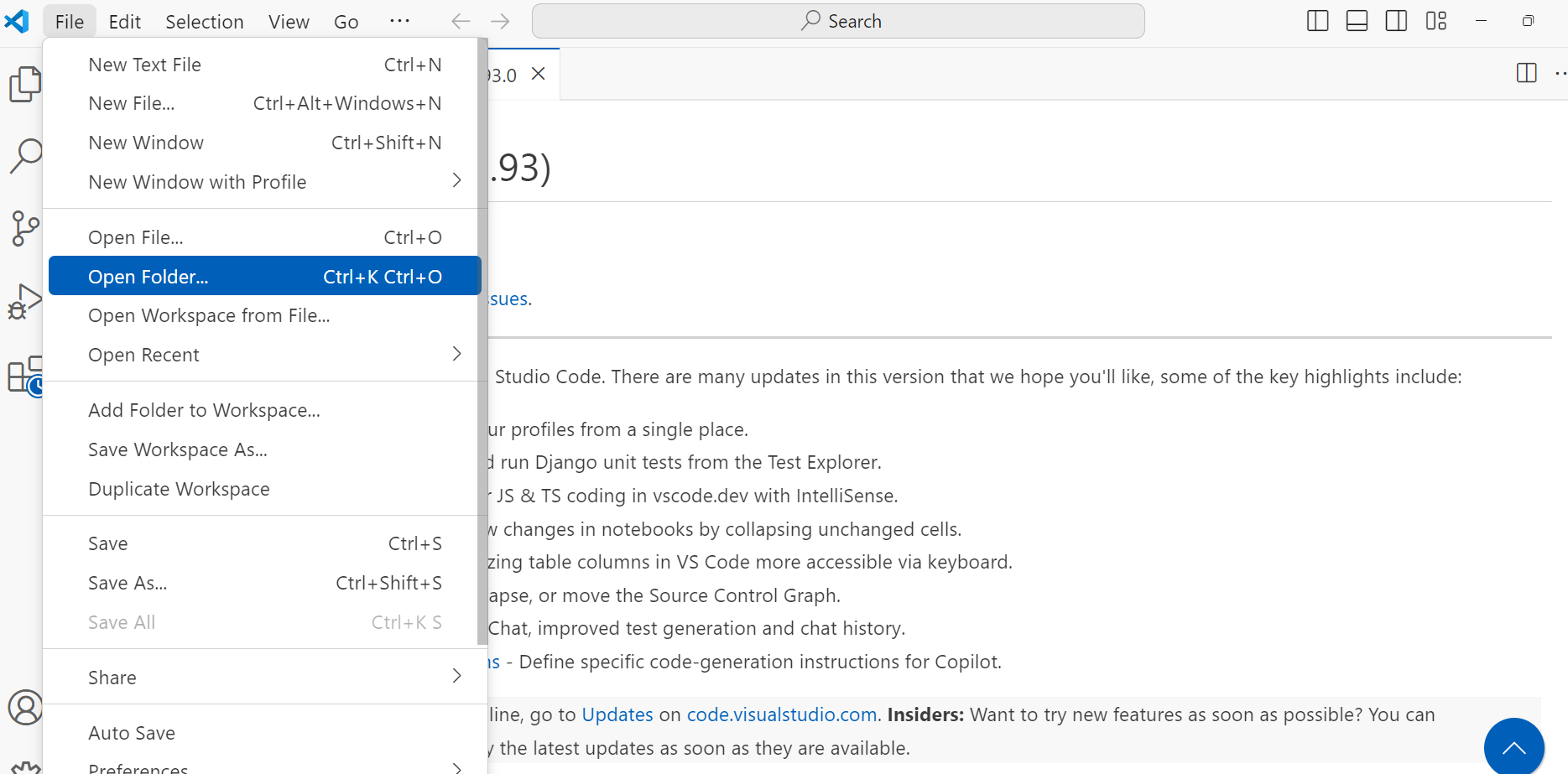
- `sendKeys()`: Simulates typing.

- `click()`: Simulates a button click.

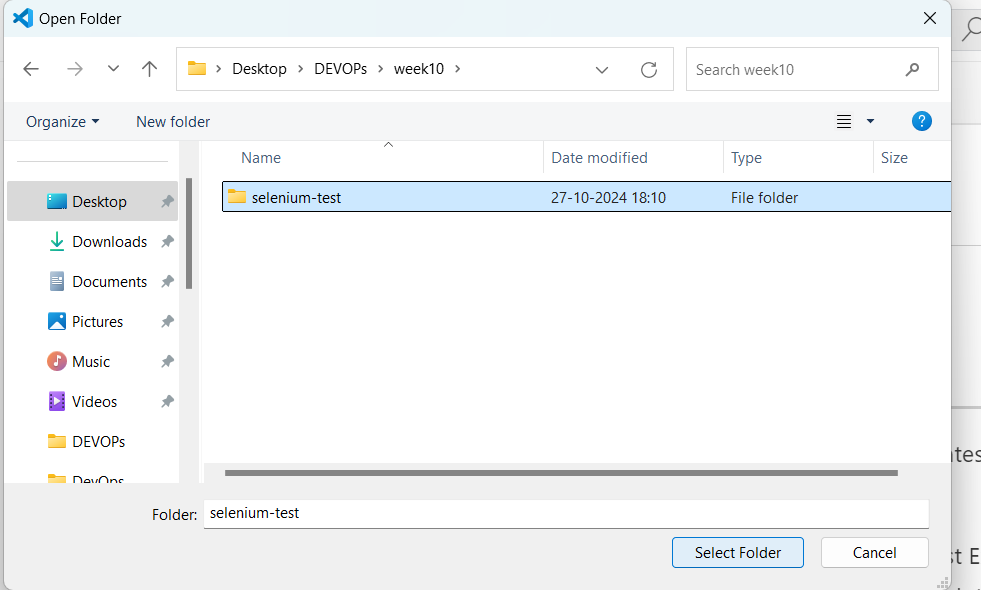
- `getTitle()`: Gets the title of the current page.

**WEEK 11. Write a Simple Program in JavaScript and Perform Testing Using Selenium**

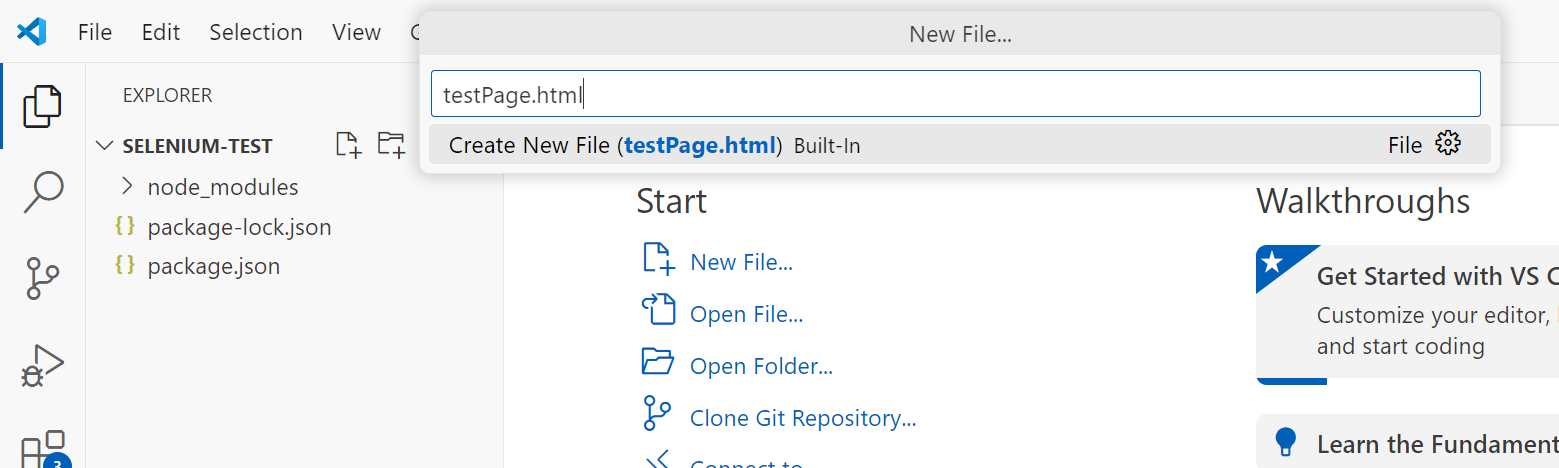
**1)Open vs code**

****

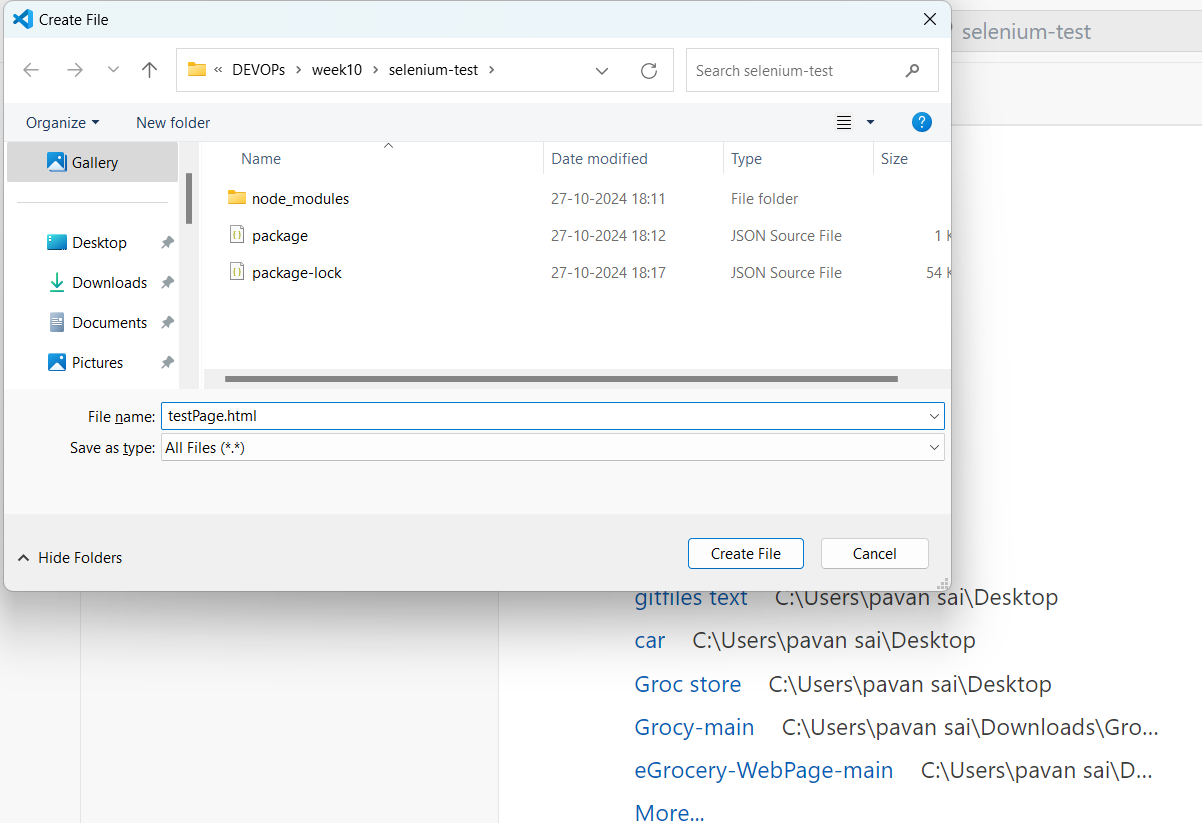
**2)Click on open folder select created folder**

****

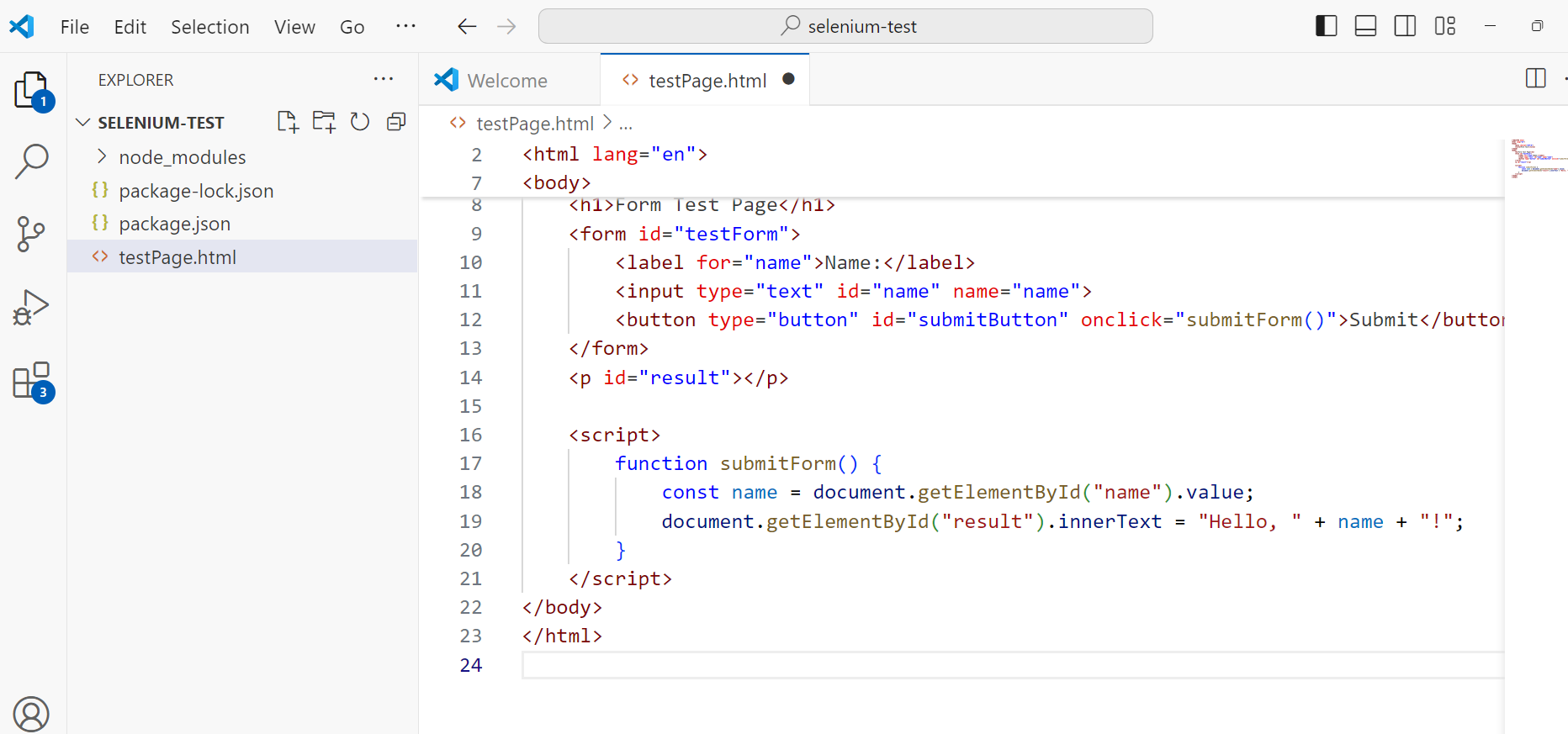
**3)Click on select folder**



**4) Create a new file in the folder testPage.html**



5) Click on **create a file**



**testPage.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Form Test</title>

</head>

<body>

<h1>Form Test Page</h1>

<form id="testForm">

<label for="name">Name:</label>

<input type="text" id="name" name="name">

<button type="button" id="submitButton" onclick="submitForm()">Submit</button>

</form>

<p id="result"></p>

<script>

function submitForm() {

const name = document.getElementById("name").value;

document.getElementById("result").innerText = "Hello, " + name + "!";

}

</script>

</body>

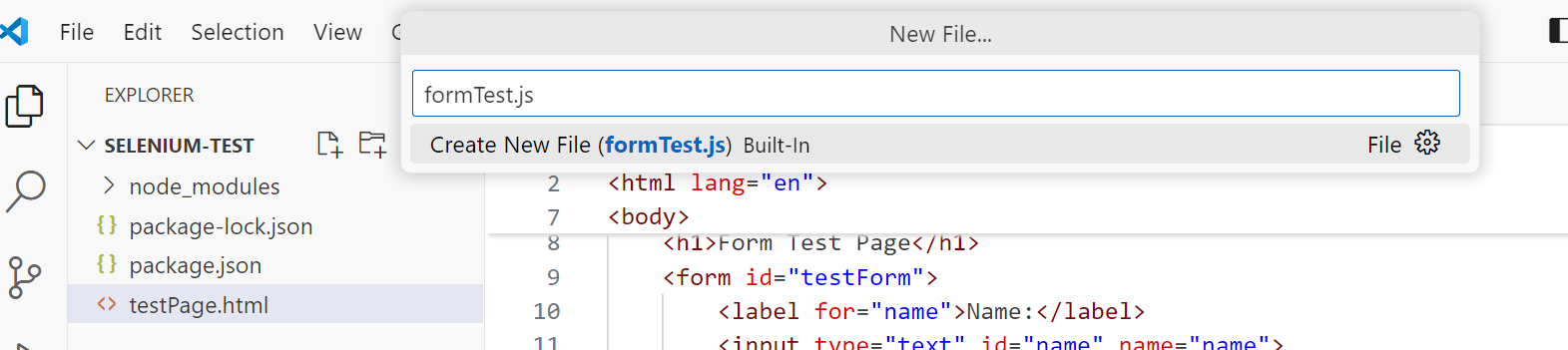
</html>

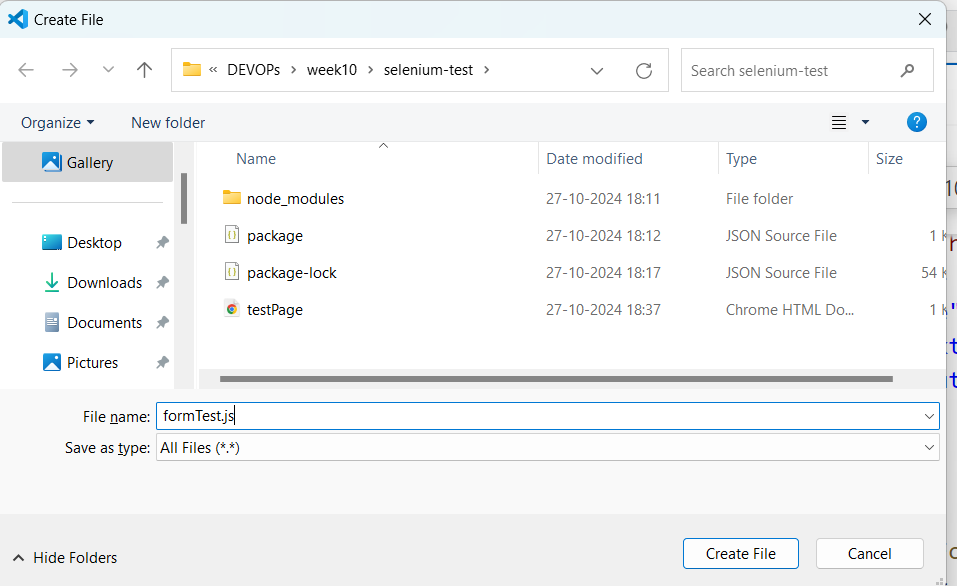
**This HTML file has:**

A text input field for entering a name.

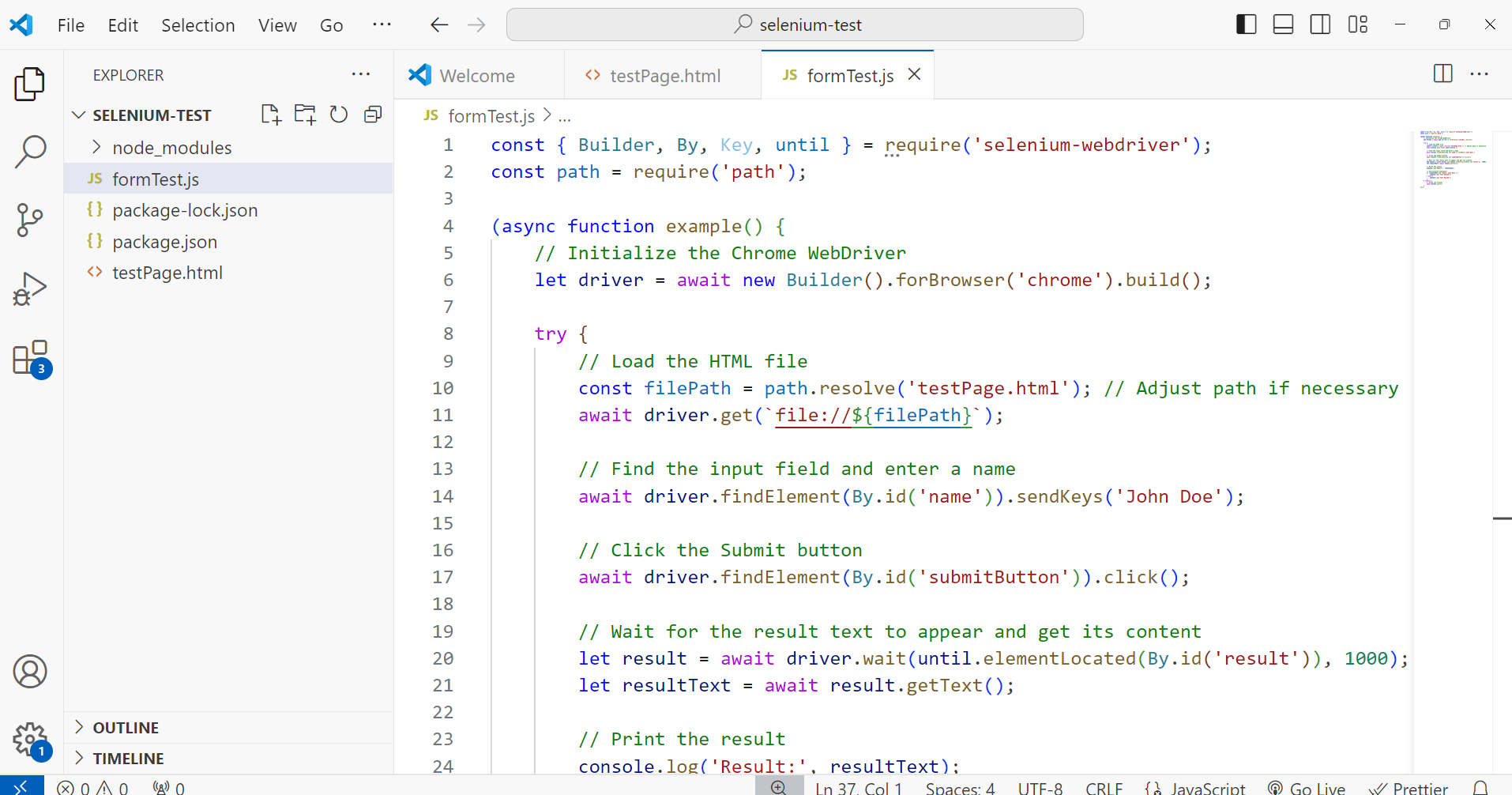
A button that, when clicked, displays a greeting message with the entered name in a <p> element.

**6) Create the JavaScript Test Script in vscode**





**7)Click on create file**

****

**formTest.js**

const { Builder, By, Key, until } = require('selenium-webdriver');

const path = require('path');

(async function example() {

// Initialize the Chrome WebDriver

let driver = await new Builder().forBrowser('chrome').build();

try {

// Load the HTML file

const filePath = path.resolve('testPage.html'); // Adjust path if necessary

await driver.get(`file://${filePath}`);

// Find the input field and enter a name

await driver.findElement(By.id('name')).sendKeys('John Doe');

// Click the Submit button

await driver.findElement(By.id('submitButton')).click();

// Wait for the result text to appear and get its content

let result = await driver.wait(until.elementLocated(By.id('result')), 1000);

let resultText = await result.getText();

// Print the result

console.log('Result:', resultText);

// Verification (optional)

if (resultText === 'Hello, John Doe!') {

console.log('Test Passed');

} else {

console.log('Test Failed');

}

} finally {

// Close the browser

await driver.quit();

}

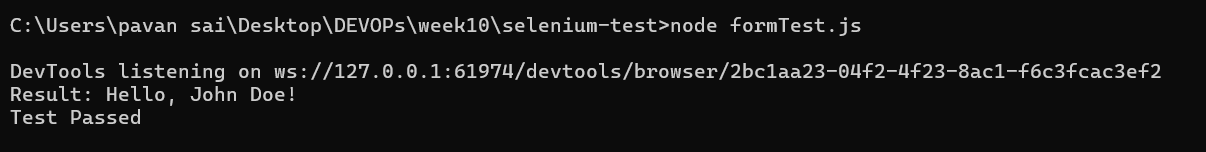
})();

**Working**

* driver.get() loads the local HTML file using the file:// protocol.
* findElement(By.id('name')).sendKeys('John Doe'); enters "John Doe" into the input field.
* findElement(By.id('submitButton')).click(); simulates clicking the submit button.
* wait(until.elementLocated(By.id('result')), 1000); waits for the result text to appear.
* Retrieves the text from the <p> element with ID result.

**8)Open command prompt and run the following command**

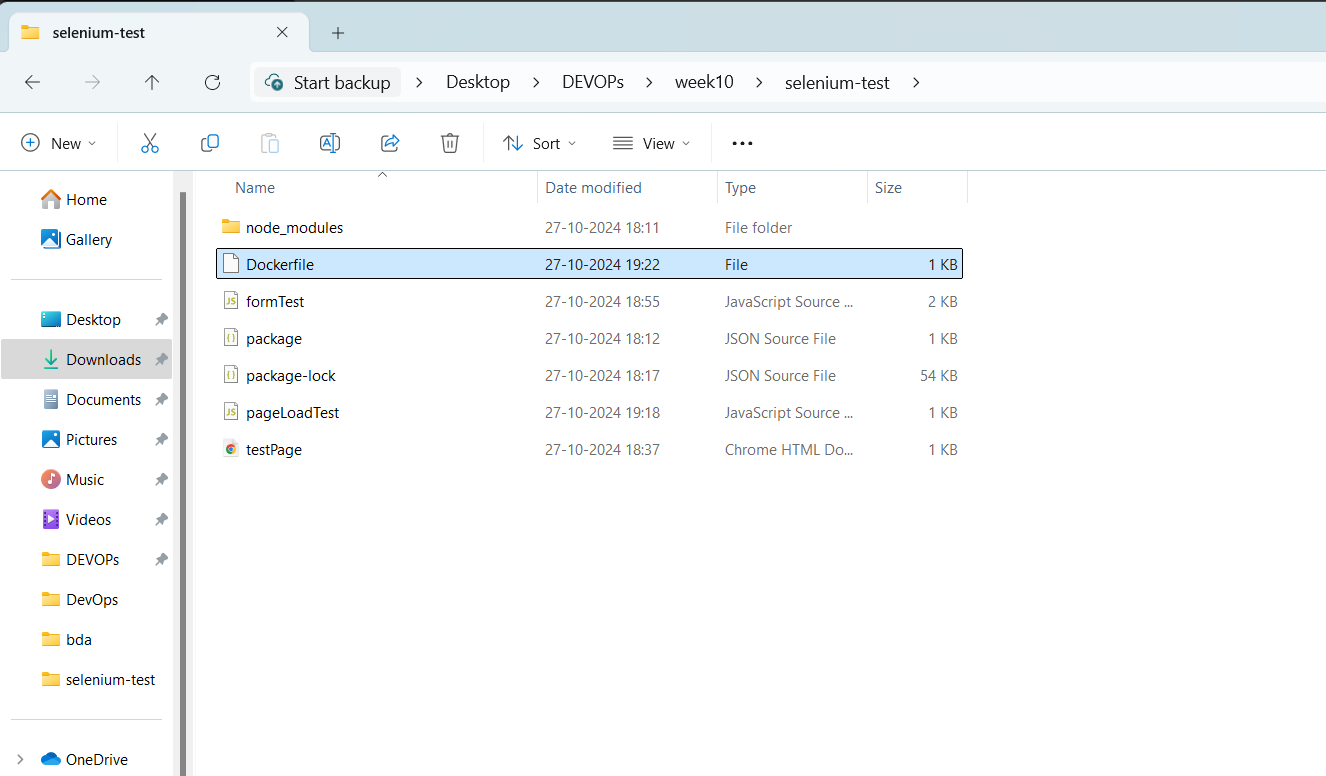
node formTest.js



**Script opens a browser.** **Navigate to html page and enters the name.**

**Week 12 Develop testcases for above containerized application using selenium**

**Create a Dockerfile in the project folder**

****

**Dockerfile**

**# Use a lightweight web server like nginx or httpd**

**FROM nginx:alpine**

**# Copy the application files to the web server's root directory**

**COPY . /usr/share/nginx/html**

**# Expose port 80**

**EXPOSE 80**

**# Start nginx server**

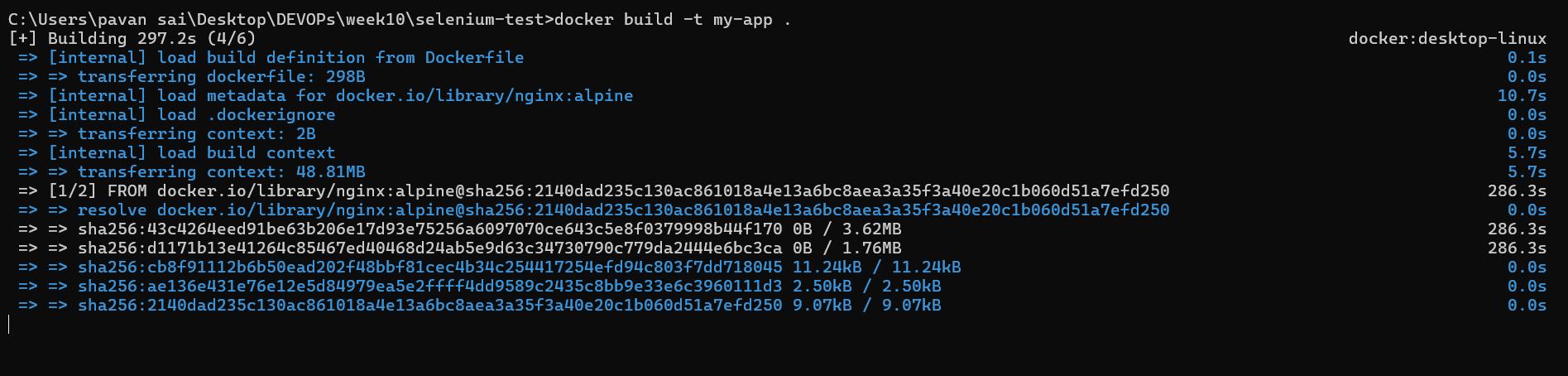
**CMD ["nginx", "-g", "daemon off;"]**

**2) Open a terminal or command prompt.**

Run the following command

**docker build -t my-app .**

This will build the Docker image and tag it as my-app.

****

**3) After run the command**

**docker run -d -p 8080:80 my-app**

**4)open VScode and write the testcases**

**3 taste cases are:**

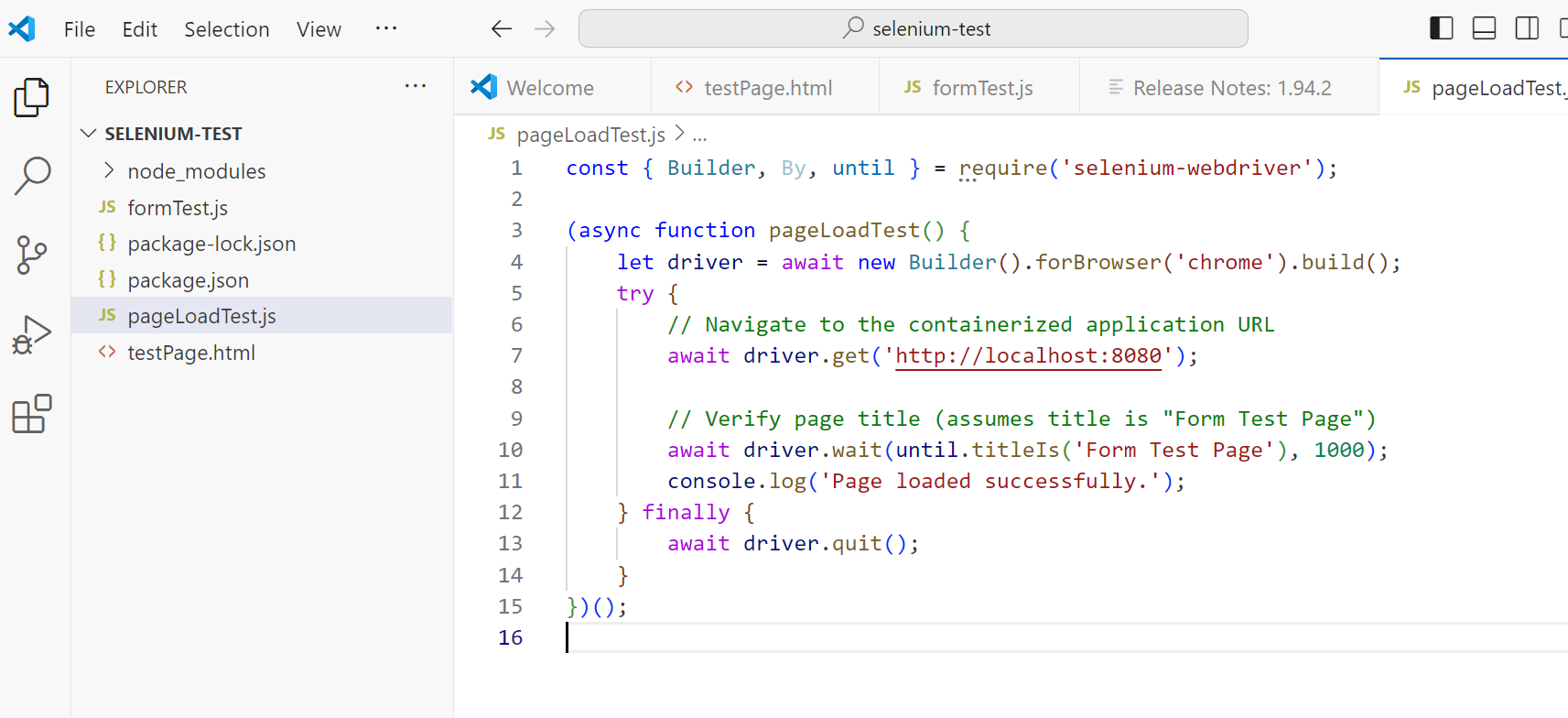
**Page Load**: Verify the page loads successfully.

**Form Submission with Input**: Submit the form with valid input and verify the response.

**Form Submission without Input**: Submit the form without input to check how it handles empty submissions.

1. **Test Case: Verify Page Loads Successfully**

Create a new file named pageLoadTest.js:



**pageLoadTest.js:**

const { Builder, By, until } = require('selenium-webdriver');

(async function pageLoadTest() {

let driver = await new Builder().forBrowser('chrome').build();

try {

// Navigate to the containerized application URL

await driver.get('http://localhost:8080');

// Verify page title (assumes title is "Form Test")

await driver.wait(until.titleIs('Form Test'), 1000);

console.log('Page loaded successfully.');

} finally {

await driver.quit();

}

})();

1. **Verify Form Input and Submission**

formSubmissionTest.js

This test fills in the form, submits it, and verifies the result.

(async function formSubmissionTest() {

let driver = await new Builder().forBrowser('chrome').build();

try {

await driver.get('http://localhost:8080');

// Fill the form with a sample name

await driver.findElement(By.id('name')).sendKeys('Alice');

// Click the submit button

await driver.findElement(By.id('submitButton')).click();

// Wait for the result and check the displayed message

let resultText = await driver.wait(until.elementLocated(By.id('result')), 1000).getText();

if (resultText === 'Hello, Alice!') {

console.log('Form submission test passed.');

} else {

console.log('Form submission test failed.');

}

} finally {

await driver.quit();

}

})();

1. **Verify Empty Form Submission**

**emptyFormTest.js**

(async function emptyFormTest() {

let driver = await new Builder().forBrowser('chrome').build();

try {

await driver.get('http://localhost:8080');

// Click the submit button without entering any data

await driver.findElement(By.id('submitButton')).click();

// Check if a specific message or behavior is shown

let resultText = await driver.wait(until.elementLocated(By.id('result')), 1000).getText();

if (resultText === 'Hello, !') {

console.log('Empty form test passed.');

} else {

console.log('Empty form test failed.');

}

} finally {

await driver.quit();

}

})();

Run the following command in command prompt

**node pageLoadTest.js**

**node formSubmissionTest.js**

**node emptyFormTest.js**