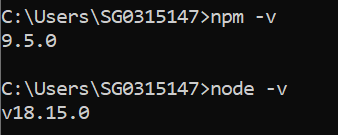
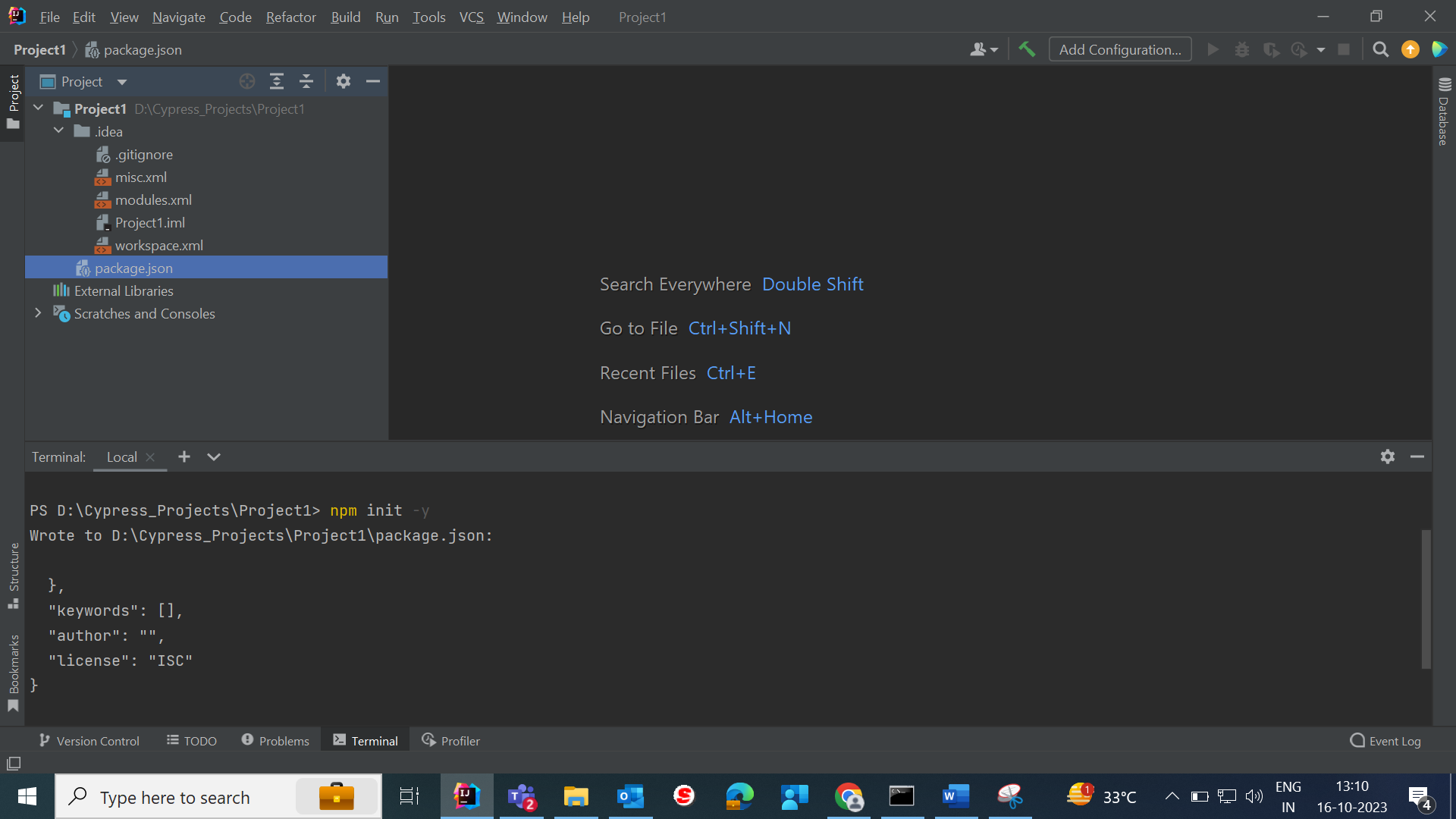
1. Install Node.js

And than verify whether its installed correctly or not.

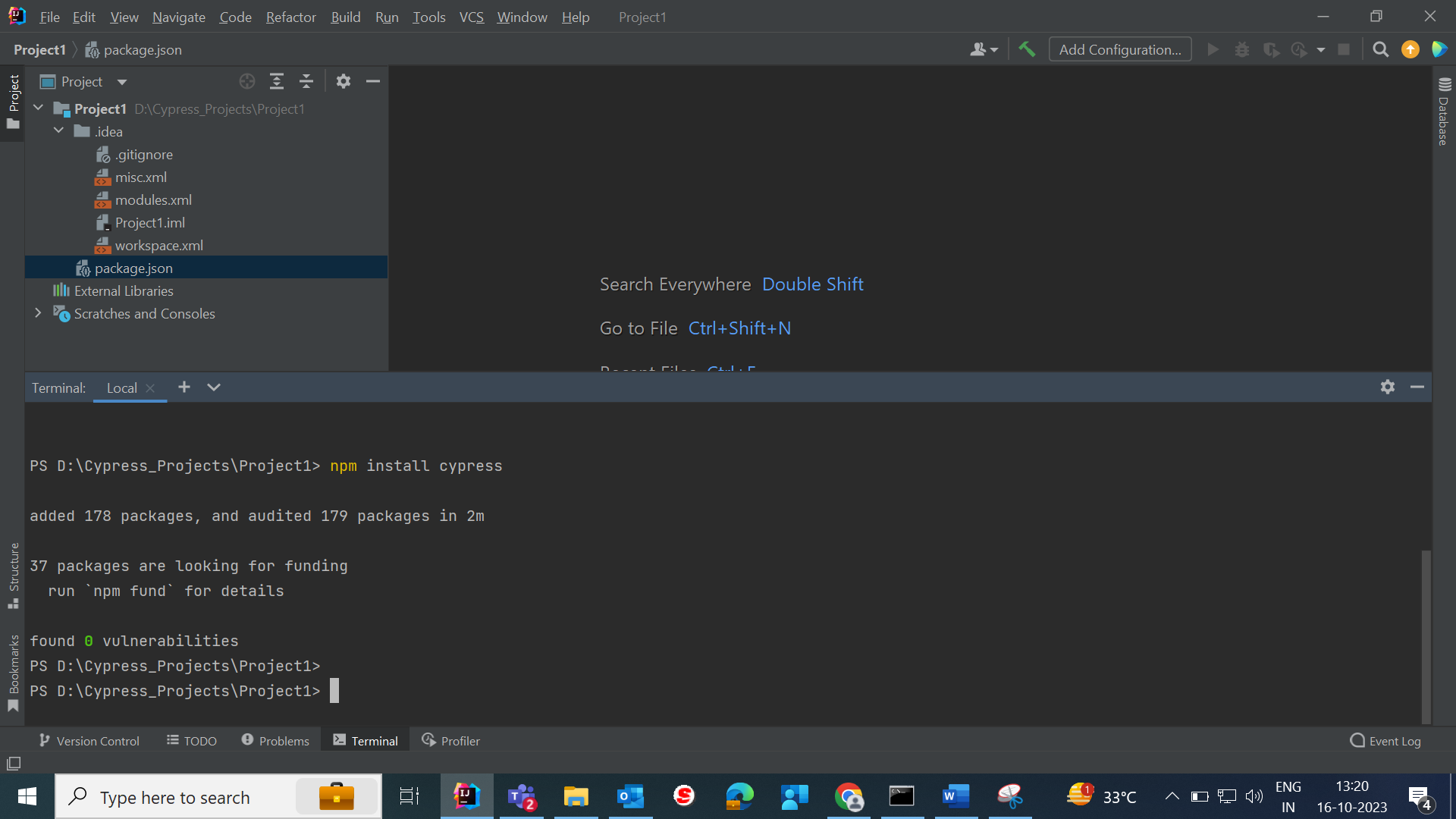


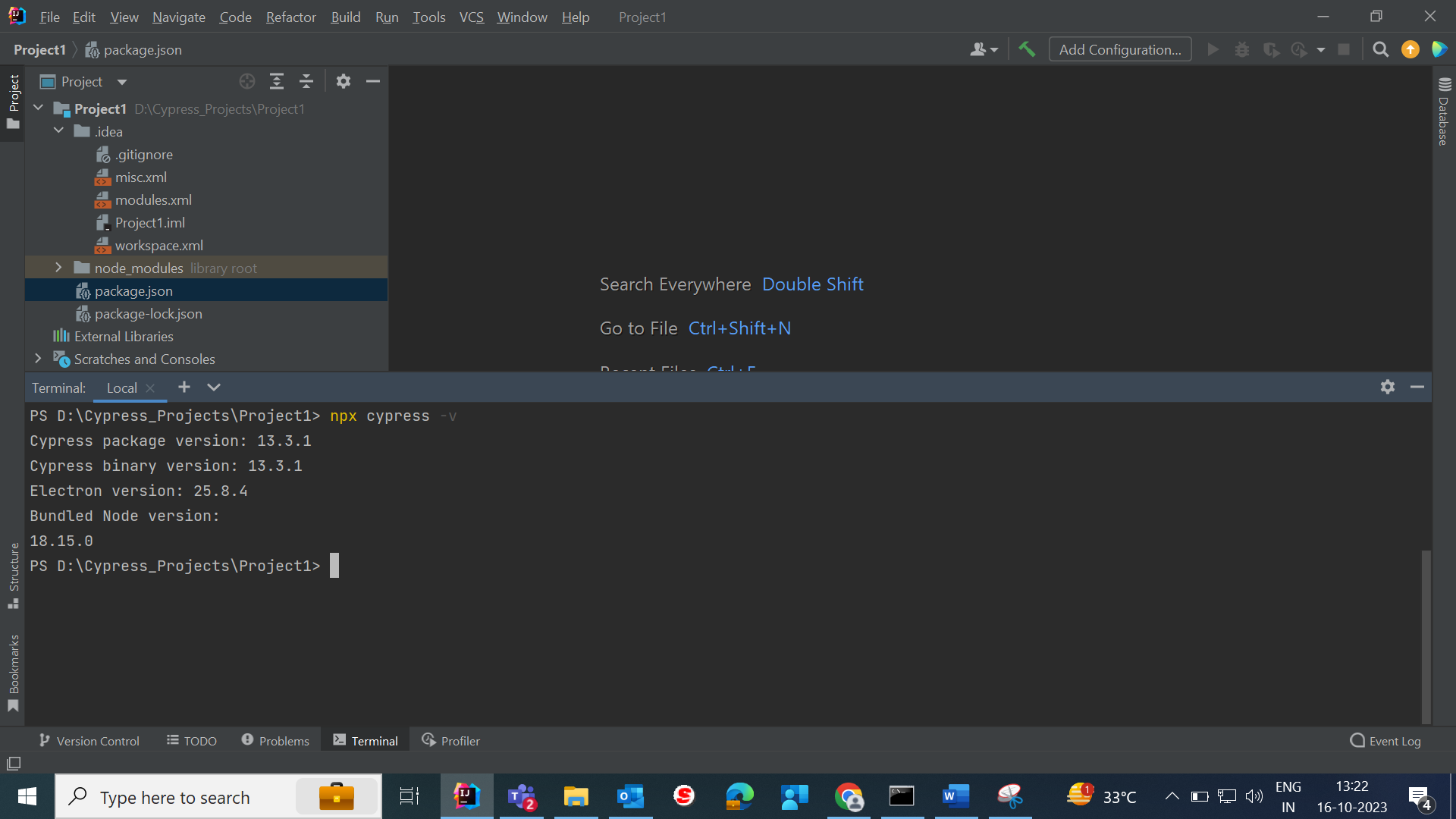
1. Install Visual Studio Code
2. Create a folder where need to create all cypress projects.
3. Now open that folder inside IDE tool.
4. Execute command – npm init -y



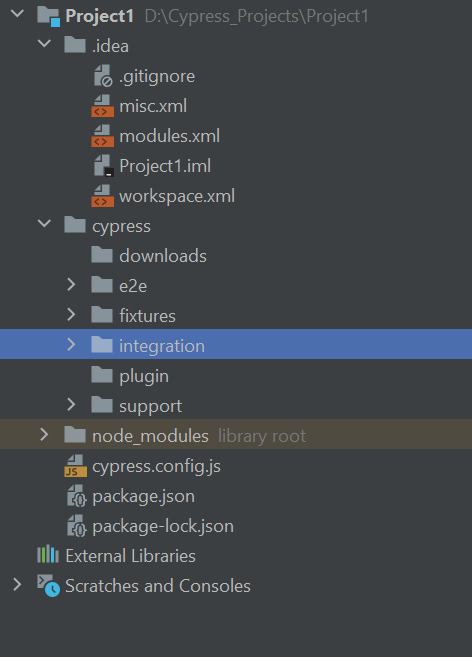
1. Now execute command – npm install cypress

To check version of Cypress 🡪 npx cypress -v





1. Open Cypress---- npx cypress open



1. Create 1st Cypress test—

Step 1:- Create file under Cypress—Integration folder.

Step 2:- At the top mention /// <reference types=”cypress”/>

It is used to only auto-complition using Cypress library

Step 3:- To write a test we need a testrunner – **mocha**

Mocha comes build-in with Cypress

Step 4:- Write the test function

Cypress Selector:-

1. Get HTML Element by ID Selector in Cypress

**ID** is an attribute of an HTML tag, which is used to find an HTML element. Using the Cypress command – **cy.get()**, you can directly pass the id with prefix **#** and get the element.

<input id="user\_email\_login" class="user\_email\_ajax">

To get the HTML element by id in Cypress, use the following command:

cy.get('#user\_email\_login')

In this command, **#** is used as a prefix to id inside **cy.get()**

Once you are able to find the HTML element, you can perform operations on the elements such as **type**, **click**, etc., as seen in the example below:

cy.get('#user\_email\_login').type('myid98788');

1. Get HTML Element by Class Selector in Cypress

Just like **ID**, **Class** is also an attribute of an HTML element, that is used as a locator or selector. Similar to how ID is directly passed with a prefix **#** using the Cypress command **cy.get()**, **class** can also be used to get the HTML element with **.** **(dot)** as a prefix inside **cy.get()**.

For example, consider code for an HTML element with **class** name **user\_email\_ajax**

<input id="user\_email\_login" class="user\_email\_ajax">

To get the HTML element by id in Cypress, use the following command:

cy.get('.user\_email\_ajax')

In this command, **. (dot)** is used as a prefix to class inside **cy.get()**

1. Get HTML Element by Tag Name Selector in Cypress

Using the Cypress command – **cy.get()**, you can directly get the element by passing the tag name.

Consider code for an HTML element with **tag** name **input**

<input id="user\_email\_login" class="user\_email\_ajax">

To get the HTML element by Tag name in Cypress, use the following command:

cy.get('input')

1. Get HTML Element by Attribute in Cypress

Cypress provides a way to get the element by attribute name. Since it supports all the different types of CSS selectors, you can pass the CSS selectors inside the **cy.get()** command to get an element.

Let’s understand this using an example:

Consider code for an HTML element with **id** name **user\_email\_login** and **class** name **user\_email\_ajax**

<input id="user\_email\_login" class="user\_email\_ajax" type="text" name="user\_login">

In the above code, there is an attribute called **name.**To get an element using an attribute in Cypress, you can simply pass CSS selector to **cy.get()**as seen below:

cy.get('input[name="user\_login"]')

However, id and class are also attributes. So, instead of using **#**, or **.** **(dot)** prefix, you can also get the element by using ID name or Class name by using ID and class as attributes.

* **Get element by ID using id as an attribute**

cy.get('input[id="user\_email\_login"]')

* **Get element by class name using class as an attribute**

cy.get('input[class="user\_email"]')

The below example demonstrates Cypress Locators with actions:

describe('Cypress Locators Demo', () => {

it('Verify Error Message', () => {

cy.visit("https://www.browserstack.com/users/sign\_in");

cy.get('#user\_email\_login').type('example@example.com')

cy.get('input[id="user\_password"]').type('Mypassword123')

cy.get('#user\_submit').click();

cy.get('.pass-input-placeholder > .input-wrapper > .error-msg > .msg-body > span').should('contain.text','Invalid password');

})

})

The above code navigates to the Browserstack Sign-in page and gets relevant locators. The following line gets an email textbox element with **ID** locator and allows the user to type the email address “**example@example.com**” in it

cy.get('#user\_email\_login').type('example@example.com')

Whereas, to get the password textbox element using the CSS attribute locators and type the password “**Mypassword123**“, the following line is used.

cy.get('input[id="user\_password"]').type('Mypassword123')

Then click on Submit button using the following command

cy.get('#user\_submit').click();

The following is the Assertion for Error Message:

cy.get('.pass-input-placeholder > .input-wrapper > .error-msg > .msg-body > span').should('contain.text','Invalid password');