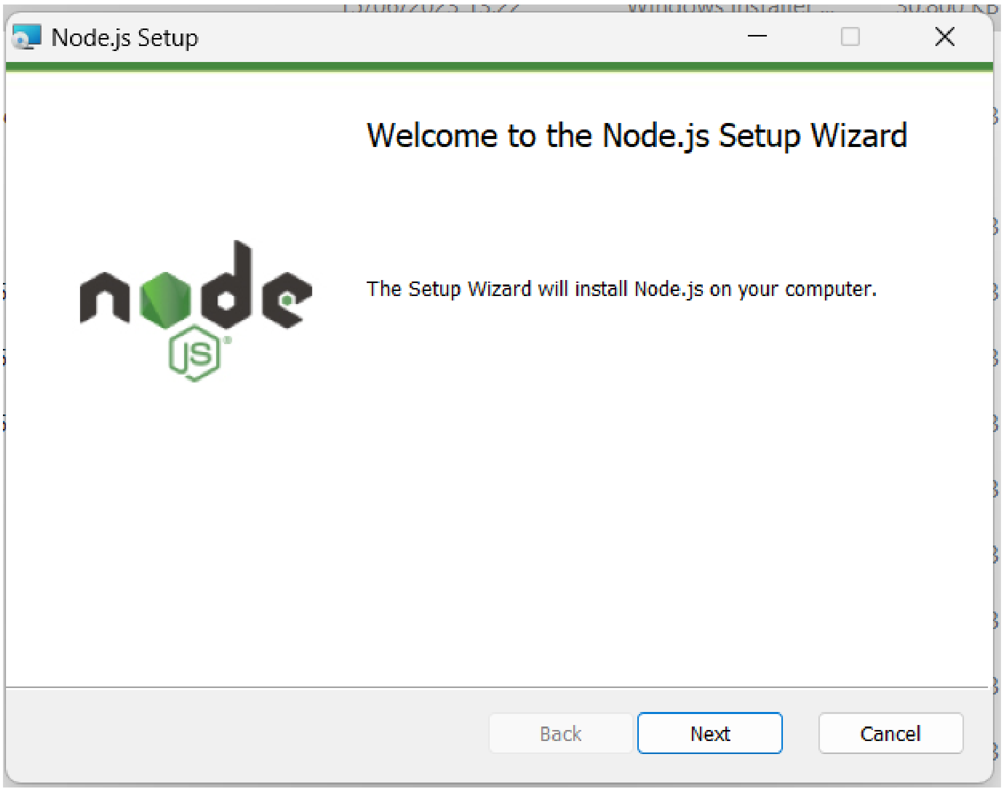
How to set up Cypress for Test Automation

Source: https://www.browserstack.com/guide/how-to-run-cypress-test-automation

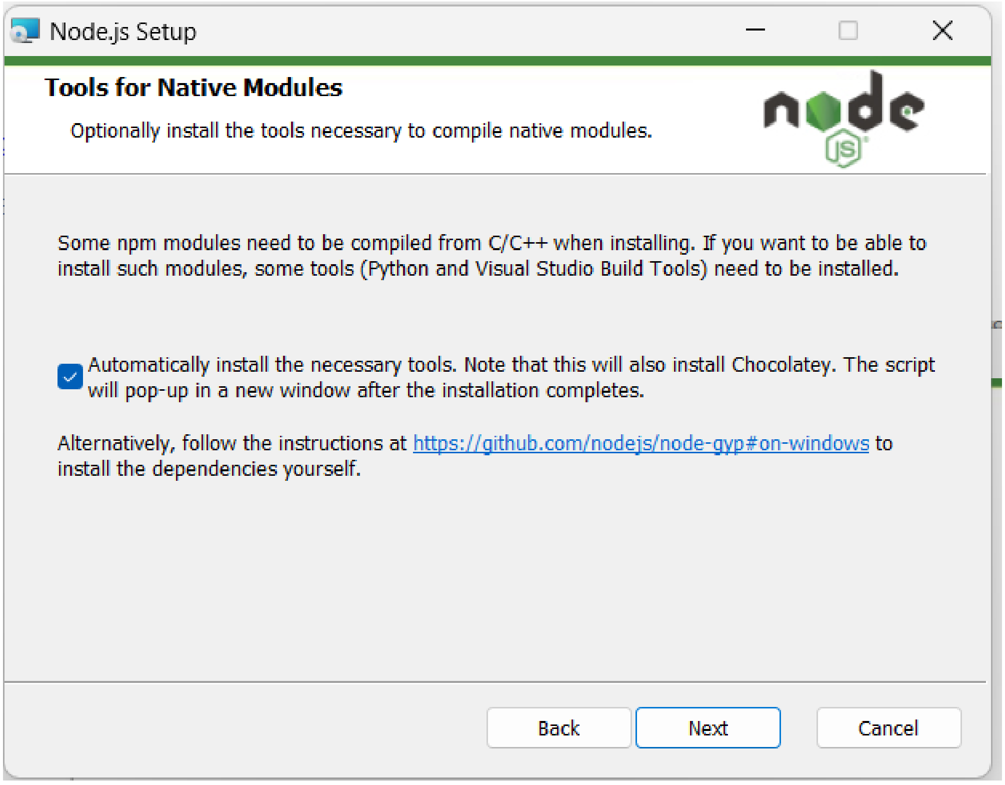
Step 1: Download and Install NodeJS

1. 

Welcome to the Node.js Setup Wizard

The Setup Wizard will install Node.js on your computer.

2. Click on Next, Next, Next, Next until



Tools for Native Modules

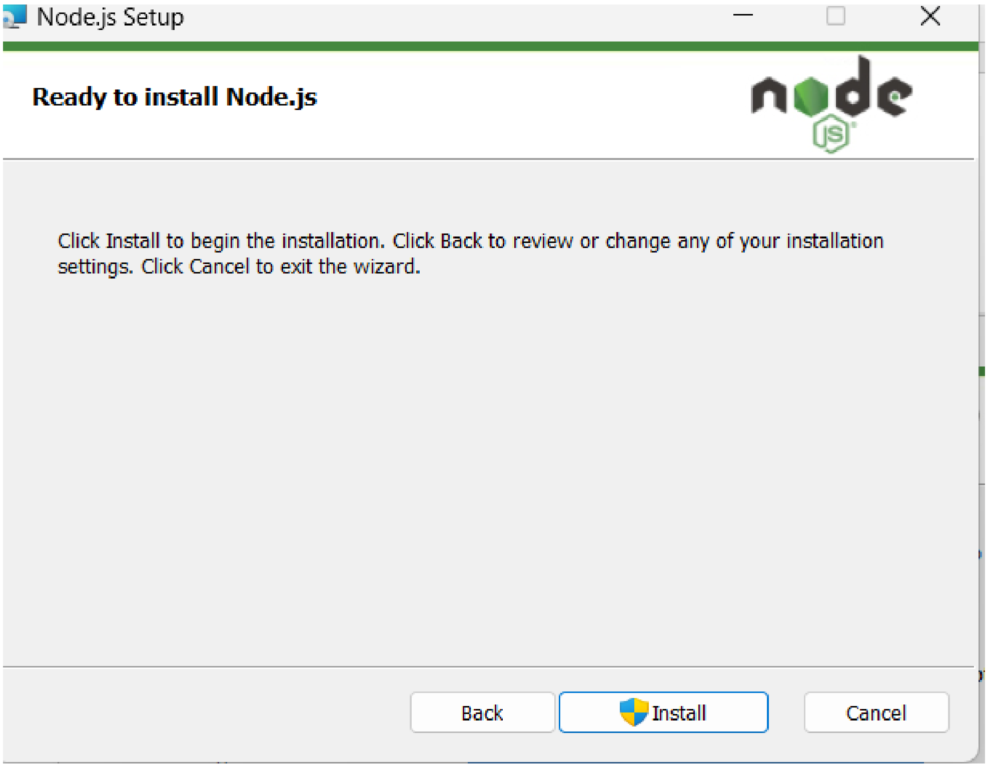
Optionally install the tools necessary to compile native modules.

Some npm modules need to be compiled from C/C++ when installing. If you want to be able to install such modules, some tools (Python and Visual Studio Build Tools) need to be installed.

☑ Automatically install the necessary tools. Note that this will also install Chocolatey. The script will pop-up in a new window after the installation completes.

Alternatively, follow the instructions at https://github.com/nodejs/node-gyp#on-windows to install the dependencies yourself.

3. Tick the box “Automatically install the necessary tools”, click on Next and
4. Click on Install



Node.js Setup

Ready to install Node.js

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Step 2: [Download and install Visual Studio Code]

Step 3: **Create Empty Project Folder in Explorer or CLI**

C:\ cypress

Step 4: **Open the folder in Visual Studio Code or Any IDE**

Go to **Visual Studio Code Menu** > Navigate to **File** > Click on **Open Folder** > Choose the newly created Folder(CypressJavascript) from **Open Folder Wizard**

Step 5: **Create package.json**

The **package.json** helps track all the packages installed for the Cypress automation framework, and it also helps to create shortcut commands to run the tests.

In order to create the **package.json**, open **Terminal** in Visual Studio Code Menu and type the command: npm init

On entering npm init in the terminal, it asks for a set of questions. Answer them or hit [Enter] [Enter] until it finishes.

Finally, it asks – Is this OK? (yes). Then, hit [Enter].

Now the file named **package.json** is automatically created in the root folder of the project.

Step 6: **Install Cypress**

Cypress is a NodeJS based automation tool, available as an npm package. Cypress can be also downloaded as an installer, but the recommended way is to install from npm.
In the root **Project Folder** (CypressJavascript) > **Terminal** > **type:** **npm install cypress --save-dev**

Step 7: **Open Cypress Window**

Once Cypress packages have been installed, Cypress, by default, configures some folders. Typically 4 folders will be created inside the Cypress folder namely plugins, support, integration, fixtures – the first time when the tester opens Cypress.

In order to open the Cypress window, use one of the following commands:

```
node ./node_modules/cypress/bin/cypress open
```

Or:

```
npx cypress open
```

On entering the above command, installation begins:
Upon successful completion of this command, the Cypress window launches, as seen below:

As mentioned earlier, this command also prepares the framework in the background.
It creates a Cypress folder in the project directory, and the Cypress folder will have subfolders like integration, fixtures, plugins, and support.
Now the Project Folder looks like below:

Let’s quickly examine these folders, why they are required:

- **e2e:** This is an important folder. All tests should sit inside this folder, because, by default, Cypress assumes that this is the test folder. Testers can create any number of subfolders inside this.
- **fixtures:** This folder helps keep data files such as **example.json,** which can be read directly inside the test scripts.
- **support:** The support folder contains common files (reusable code, global variables, etc.) that need to be accessed globally inside the framework.