

# Node.js Setup Win

## Node/NPM/Validator Setup - Windows

## Command Line Reference

### Key Controls/Shortcuts

<b>Enter/Return</b>	Executes the current command
<b>Up/Down Arrow</b>	Auto-fills previously/next run commands
<b>Control+C</b>	Stops the process currently running
<b>Tab</b>	Autocomplete a file or directory name

### File System Commands

<b>ls</b>	Lists all of the files/folders inside of the current directory
<b>cd</b>	Changes directory to the folder you specify (not including the <>)
<b>cd ..</b>	Goes up one to the parent of the current directory
<b>pwd</b>	Displays the full path to the current directory

#### [Return to Top](#)

<https://courses.cs.washington.edu/courses/cse154/20au/resources/assets/node/windows/#nodenpmvalidator-setup---windows>

## 1) Installing Node.js and NPM

### Preface/Explanation

Programming languages are just like real languages. They are used to communicate ideas in a structured way. In this course, we will be using the JavaScript programming language to both communicate to your browser how to handle the behavior of your website and to communicate what we want our server to do and respond to requests with.

On the client-side, your JavaScript is interpreted by your browser, so it knows how to respond to user events like button clicks and pages loads. On the server-side, your JavaScript is interpreted by a piece of software called Node.js, so that it knows how to respond to user requests over the network.

This includes responding with the client-side files (HTML/CSS/Client-side JavaScript) for the client to

display,

or responding with information stored or calculated on the server. While developing for this course, we will be running both the client-side browser and the server on the same machine (your computer). (We will go more in depth into the server-side later on in the course).

There are a wide array of community-made modules that expand upon the default features of Node.js.

We will be using some of these modules in this class. In order to install and manage these modules, we will use a program called npm that is included when installing Node.js. Follow the steps below to install both Node.js and npm.

## Steps

1. Start off by downloading the the LTS (version number  $\geq 14.15.0$ ) Node.js/NPM .msi installer from <https://nodejs.org/en/download/> \_ [\(https://nodejs.org/en/download/\)](https://nodejs.org/en/download/). Then, run the installer to start the installation process.

**Downloads**  
Latest LTS Version: 12.13.0 (includes npm 6.12.0)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

1) **LTS** Recommended For Most Users

2) **Current** Latest Features

**Windows Installer** (node-v12.13.0-x86.msi)

**macOS Installer** (node-v12.13.0.pkg)

**Source Code** (node-v12.13.0.tar.gz)

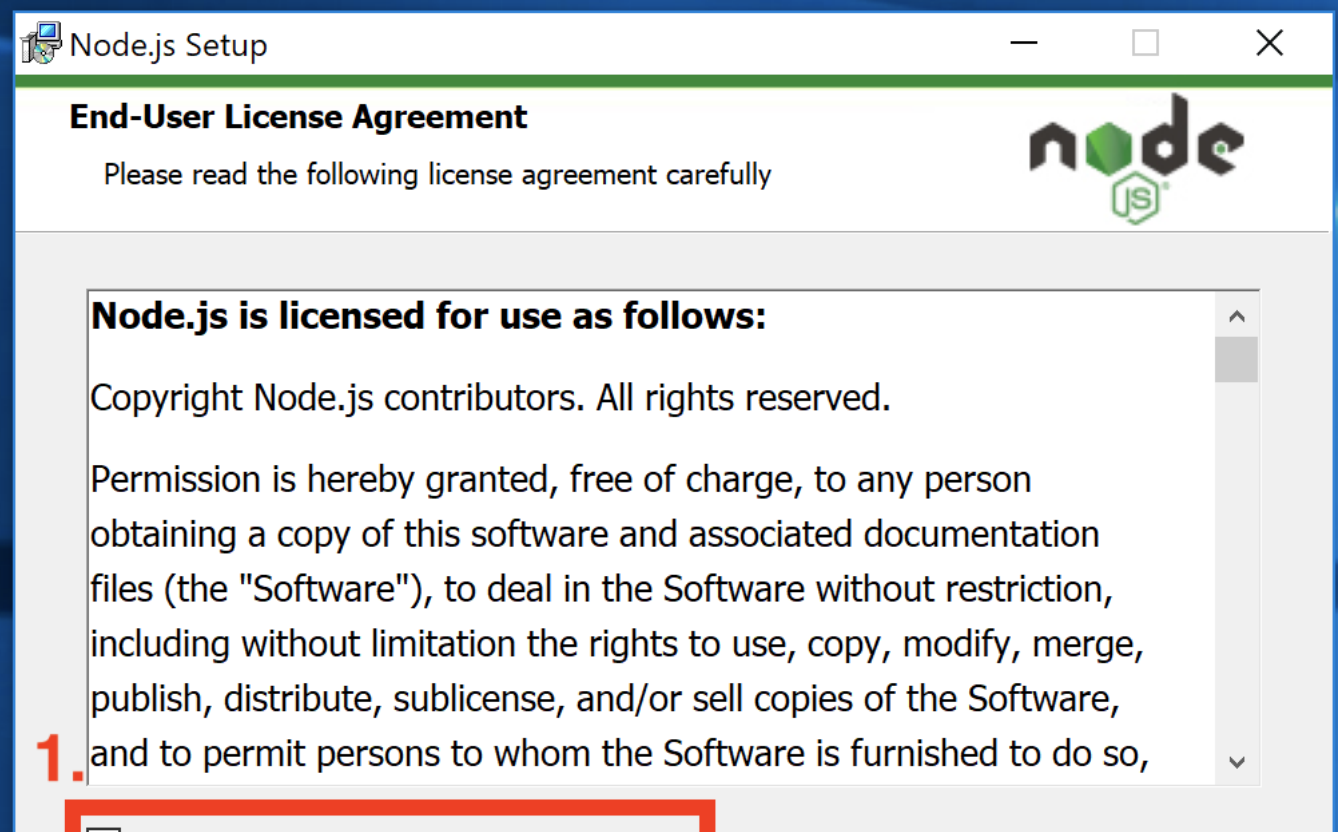
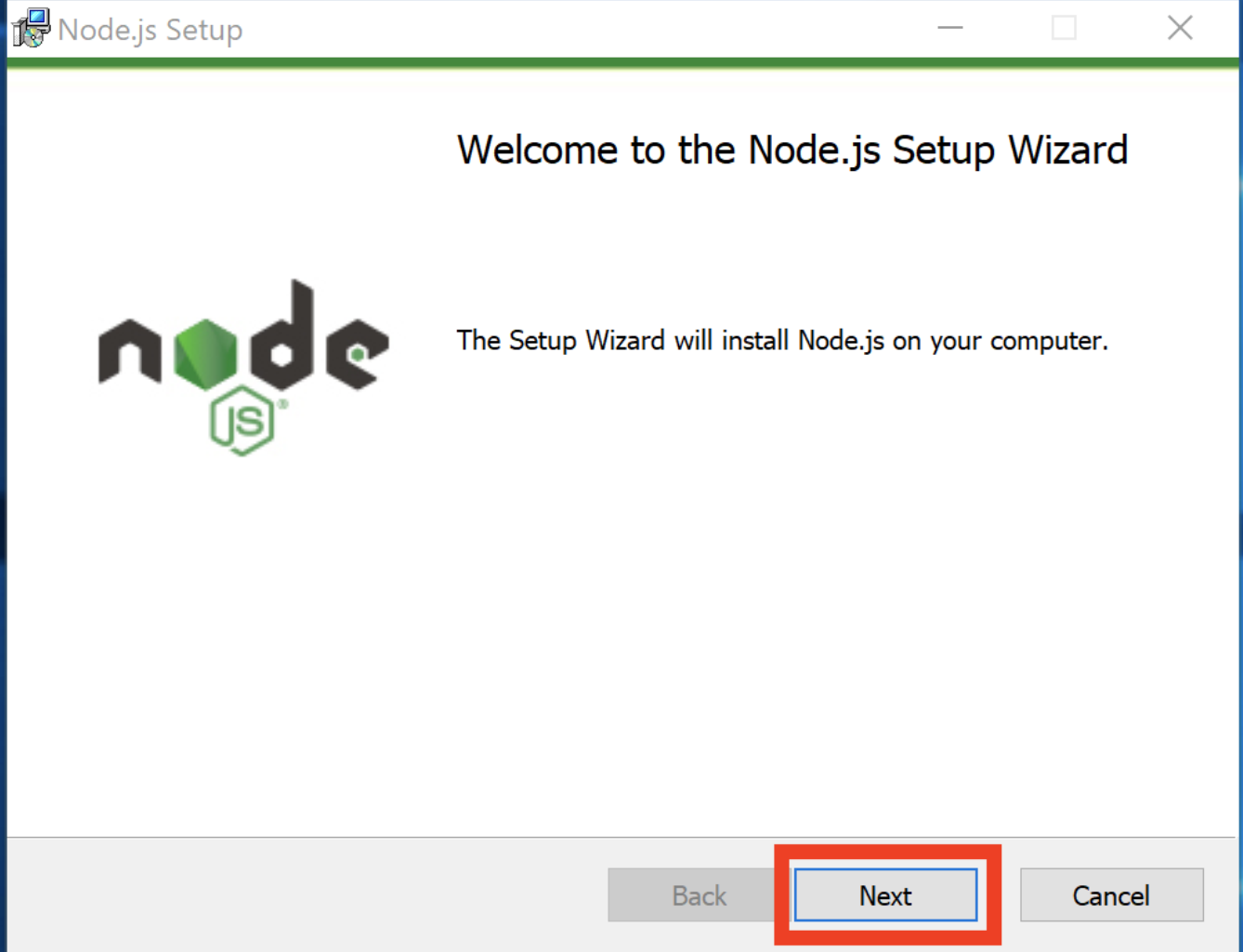
Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit	
macOS Binary (.tar.gz)	64-bit	
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v12.13.0.tar.gz	

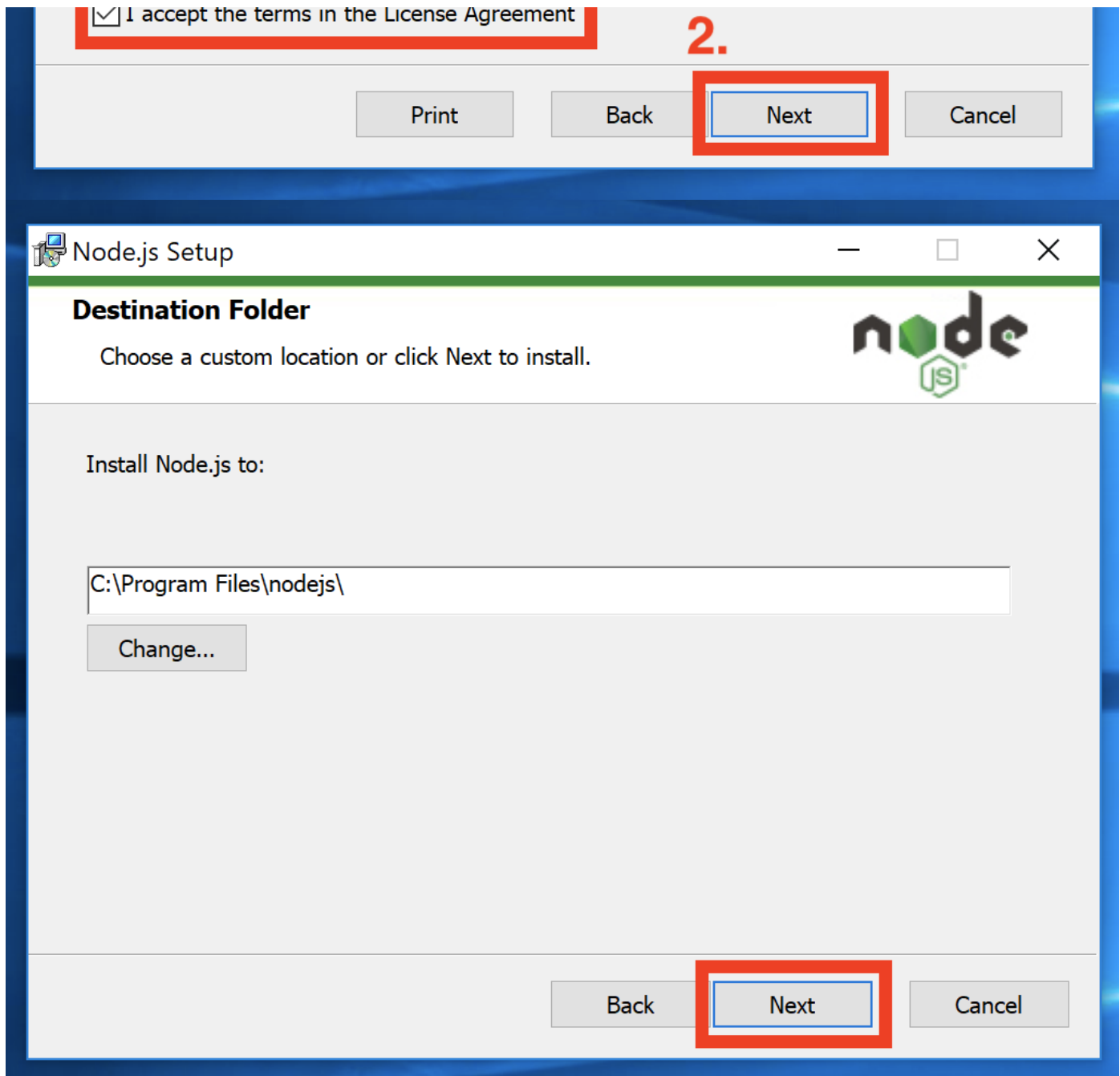
**Additional Platforms**

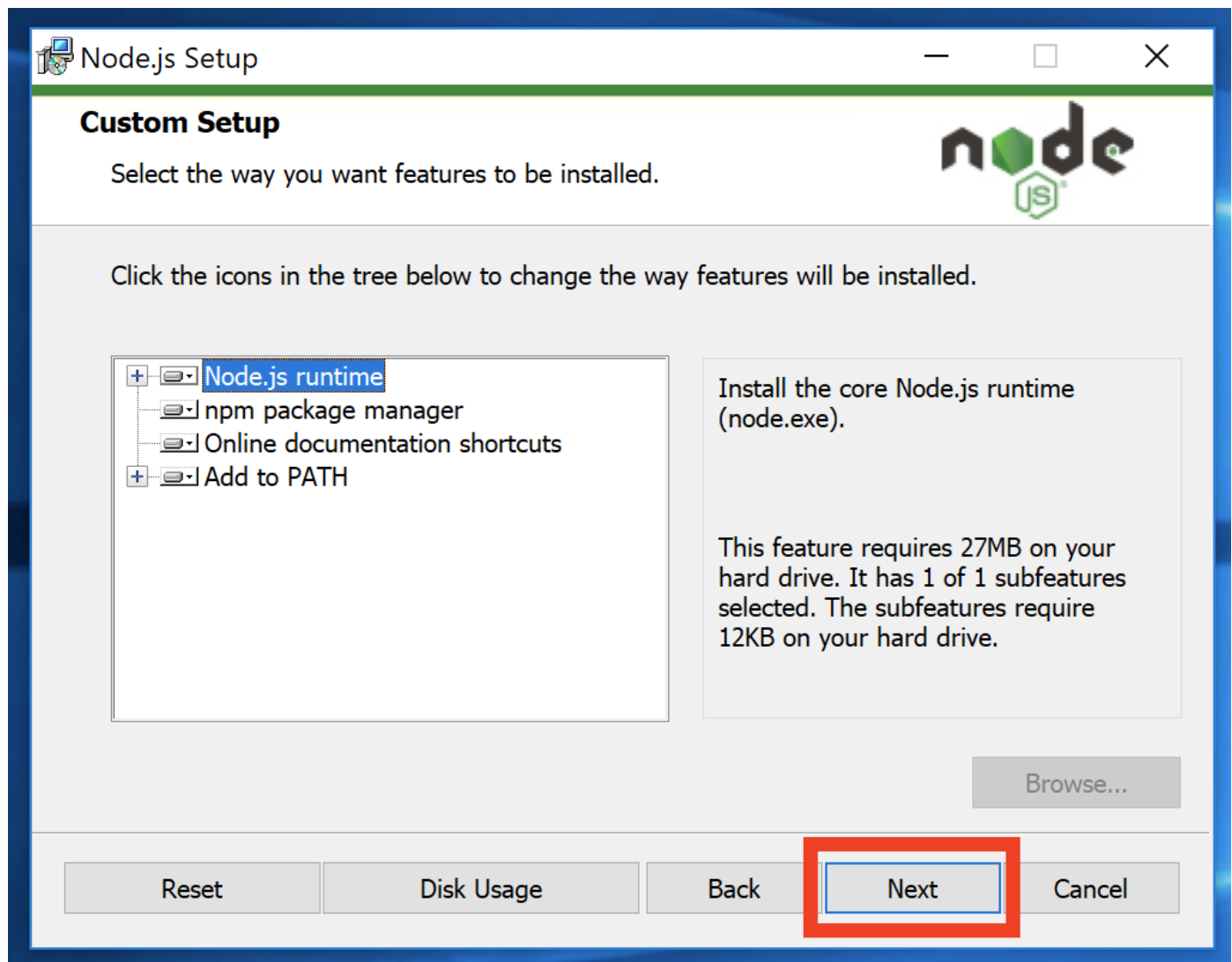
SmartOS Binaries	64-bit
Docker Image	Official Node.js Docker Image

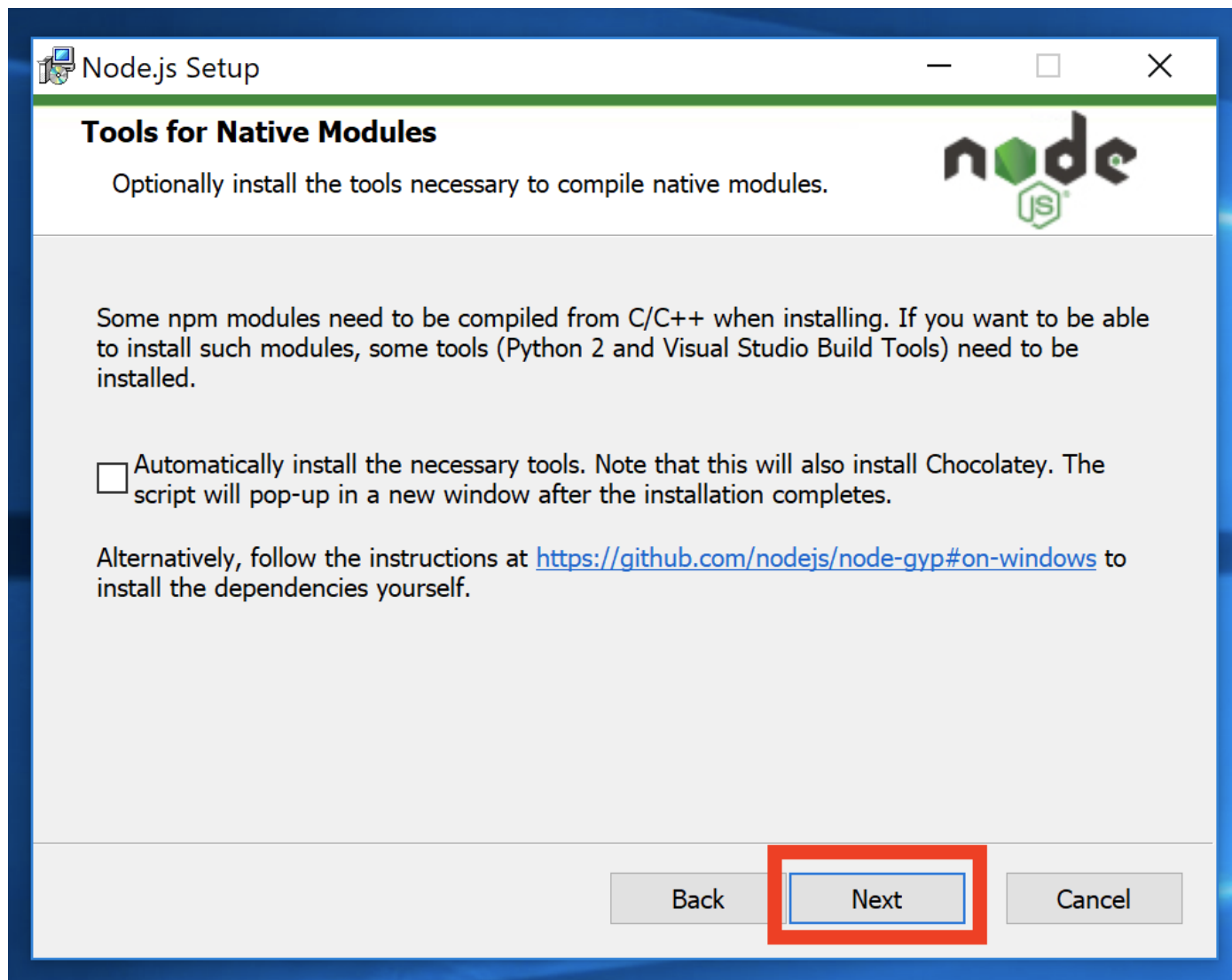
<https://nodejs.org/dist/v12.13.0/node-v12.13.0-x86.msi>

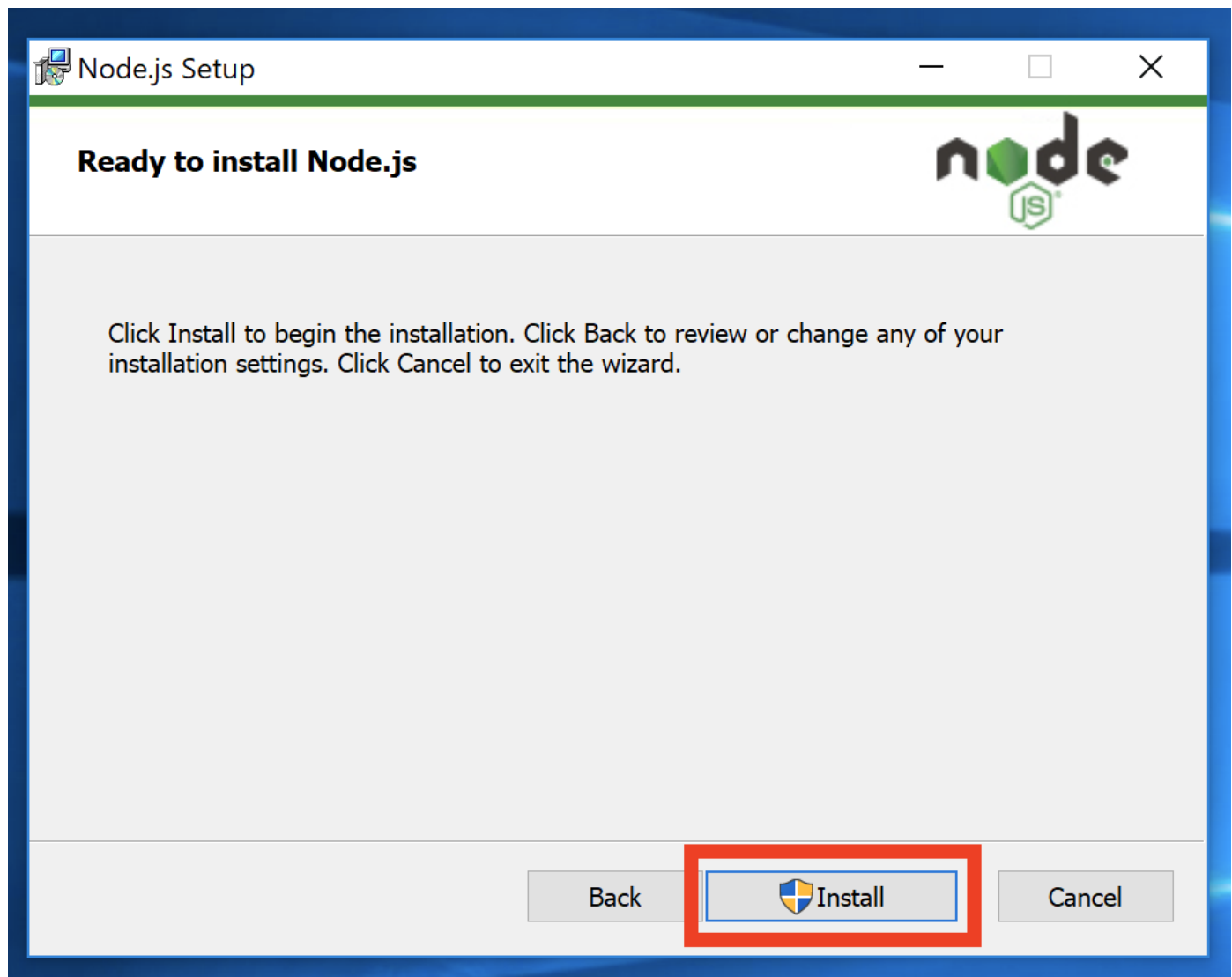
2. Go through the prompts hitting next / accept / finish each time with the default options.

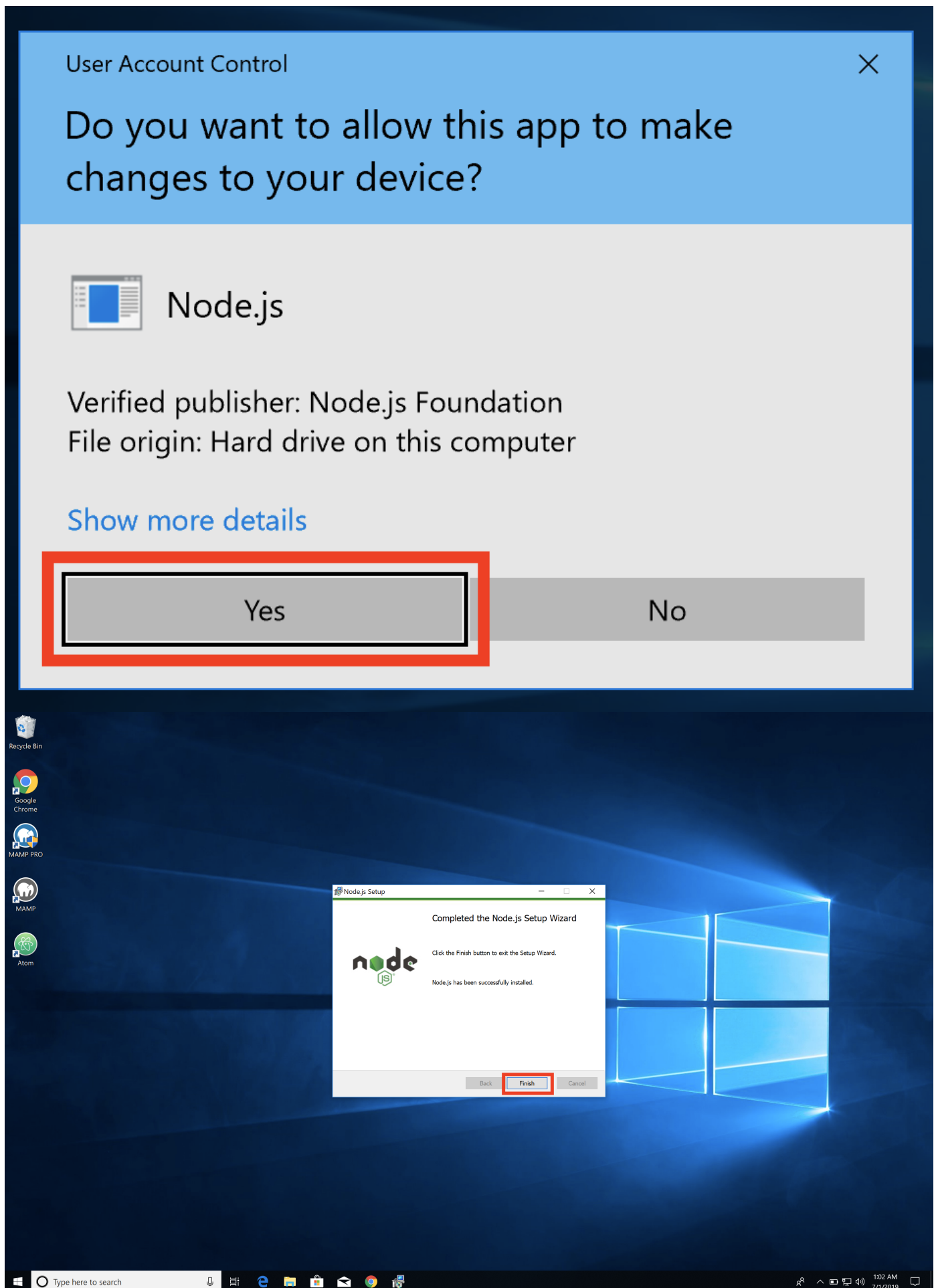






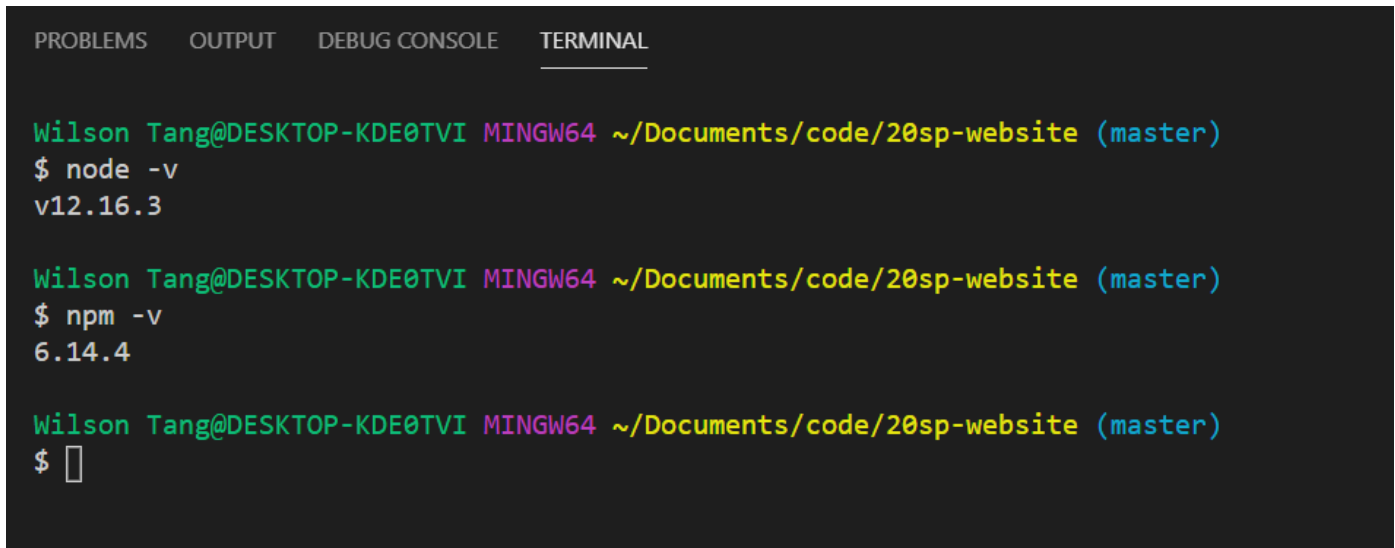








- Next, we make sure Node has installed correctly. Open Git Bash on VSCode or Git Bash regularly. You should have this already set up from the beginning of the quarter. Refer back to the guide [here](https://courses.cs.washington.edu/courses/cse154/20au/resources/assets/vscode-git-tutorial/windows/index.html) [.\(https://courses.cs.washington.edu/courses/cse154/20au/resources/assets/vscode-git-tutorial/windows/index.html\)](https://courses.cs.washington.edu/courses/cse154/20au/resources/assets/vscode-git-tutorial/windows/index.html) if not.
- Then, run both `node -v` and `npm -v` to see the version numbers and check that both commands are installed and working.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ node -v
v12.16.3

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ npm -v
6.14.4

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$
```

## 2) Installing Global Node Modules

### Preface/Explanation

Node modules, by default, are installed locally within a particular project, so they are only accessible

and importable within a particular Node project directory. However, some modules are able to be installed globally

which can allow them to create useful command-line commands. Here are a few global modules used in this course:

<code>http-server</code>	Allows you to quickly run a local server that hosts the files inside of the current directory you run the <code>http-server</code> command in.
<code>nodemon</code>	A wrapper command that acts like if you run <code>node</code> , but it detects any changes you have made to your files, and restarts the <code>node</code> process so your changes are immediately reflected without manually restarting.
<code>eslint</code>	Javascript linter commandline module that we use to run the local linters.
<code>stylelint</code>	CSS linter commandline module that we use to run the local linters.
<code>@linthtml/linthtml</code>	HTML linter commandline module that we use to run the local linters.

# Steps

1. Run the normal npm installation command for each module, but with a `-g` flag to tell npm that you want the modules installed globally. If any of these fail, try running Git Bash "as an administrator".

- `npm install -g http-server`
- `npm install -g nodemon`
- `npm install -g eslint`
- `npm install -g stylelint`
- `npm install -g @linthtml/linthtml`

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ npm install -g http-server
C:\Users\Wilson Tang\AppData\Roaming\npm\http-server -> C:\Users\Wilson Tang\AppData\Roaming\npm\node_modules\http-server\bin\http-server
C:\Users\Wilson Tang\AppData\Roaming\npm\hs -> C:\Users\Wilson Tang\AppData\Roaming\npm\node_modules\http-server\bin\http-server
+ http-server@0.12.3
added 23 packages from 35 contributors in 1.834s

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ npm install -g nodemon
C:\Users\Wilson Tang\AppData\Roaming\npm\nodemon -> C:\Users\Wilson Tang\AppData\Roaming\npm\node_modules\nodemon\bin\nodemon.js

> nodemon@2.0.3 postinstall C:\Users\Wilson Tang\AppData\Roaming\npm\node_modules\nodemon
> node bin/postinstall || exit 0

Love nodemon? You can now support the project via the open collective:
> https://opencollective.com/nodemon/donate

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@~2.1.2 (node_modules\nodemon\node_modules\chokidar\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.1.3: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ nodemon@2.0.3
added 120 packages from 54 contributors in 6.609s

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ 

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ npm install -g eslint
C:\Users\Wilson Tang\AppData\Roaming\npm\eslint -> C:\Users\Wilson Tang\AppData\Roaming\npm\node_modules\eslint\bin\eslint.js
+ eslint@6.8.0
added 133 packages from 83 contributors in 5.425s

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ npm install -g stylelint
C:\Users\Wilson Tang\AppData\Roaming\npm\stylelint -> C:\Users\Wilson Tang\AppData\Roaming\npm\node_modules\stylelint\bin\stylelint.js
+ stylelint@13.3.3
added 288 packages from 194 contributors in 8.295s

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ npm install -g @linthtml/linthtml
C:\Users\Wilson Tang\AppData\Roaming\npm\linthtml -> C:\Users\Wilson Tang\AppData\Roaming\npm\node_modules\@linthtml\linthtml\bin\linthtml.js
+ @linthtml/linthtml@0.3.2
added 156 packages from 153 contributors in 4.466s

Wilson Tang@DESKTOP-KDE0TVI MINGW64 ~/Documents/code/20sp-website (master)
$ 

```

## 3) Installing Local Linters in VSCode

### Preface/Explanation

Here we will be installing VSCode extensions and linter rules so that VSCode will display whenever your code does not lint and highlight any issues. **Note:** This is for your convenience, and you should always default to the Gitlab/Gradescope output over this local linter output.

## Steps

1. Install the VSCode extensions by following each of these links and clicking install. It should open up VSCode and install the linter extensions.
  - LintHTML (HTML): <https://marketplace.visualstudio.com/items?itemName=kamikillerto.vscode-linthtml> [\\_\(https://marketplace.visualstudio.com/items?itemName=kamikillerto.vscode-linthtml\)](https://marketplace.visualstudio.com/items?itemName=kamikillerto.vscode-linthtml)
  - ESLint (JS): <https://marketplace.visualstudio.com/items?itemName=dbaeumer.vscode-eslint> [\\_\(https://marketplace.visualstudio.com/items?itemName=dbaeumer.vscode-eslint\)](https://marketplace.visualstudio.com/items?itemName=dbaeumer.vscode-eslint)
  - Stylelint (CSS): <https://marketplace.visualstudio.com/items?itemName=stylelint.vscode-stylelint> [\\_\(https://marketplace.visualstudio.com/items?itemName=stylelint.vscode-stylelint\)](https://marketplace.visualstudio.com/items?itemName=stylelint.vscode-stylelint)