

PRIMER PROYECTO

typeScript



Contenido

Instalación de NODE.js:.....	3
Instalación de visual Studio Code:.....	8
Instalación Global de TypeScript:	9
Inicialización de un proyecto:.....	10
Instalación de un proyecto ya existente:.....	16

Instalación de NODE.js:

En primer lugar accedemos a la siguiente página:

<https://nodejs.org/es/>

Y descargamos el siguiente archivo:



Node.js® es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome.

New security releases to be made available October 12th, 2021

Descargar para Windows (x64)

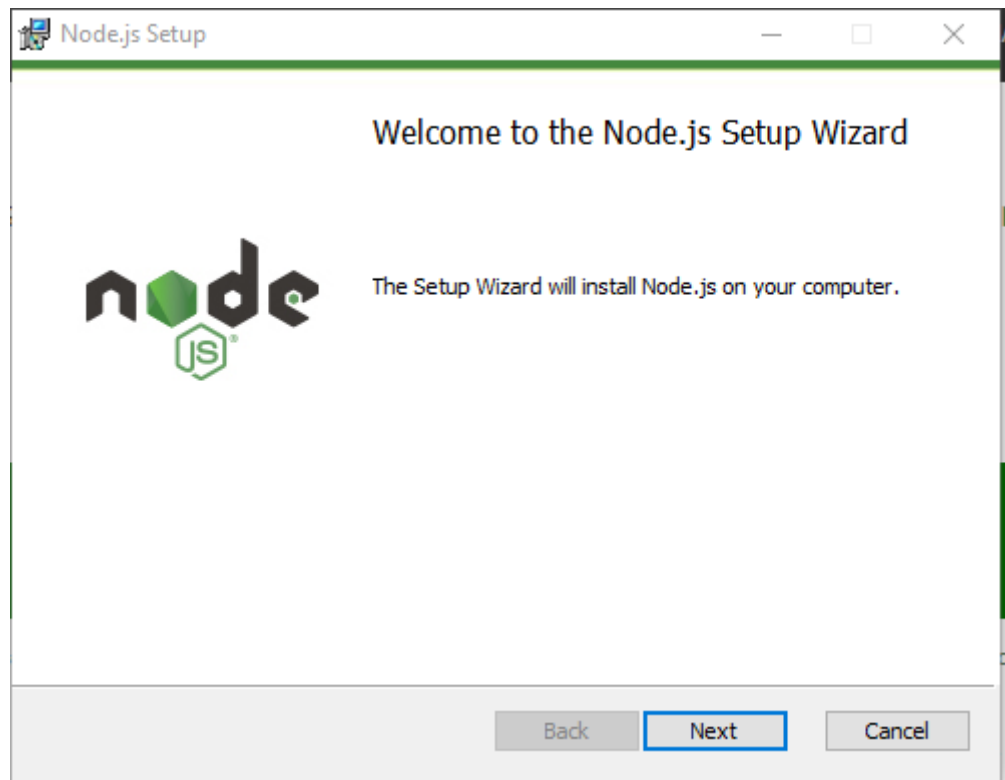
14.18.0 LTS
Recomendado para la mayoría

16.10.0 Actual
Últimas características

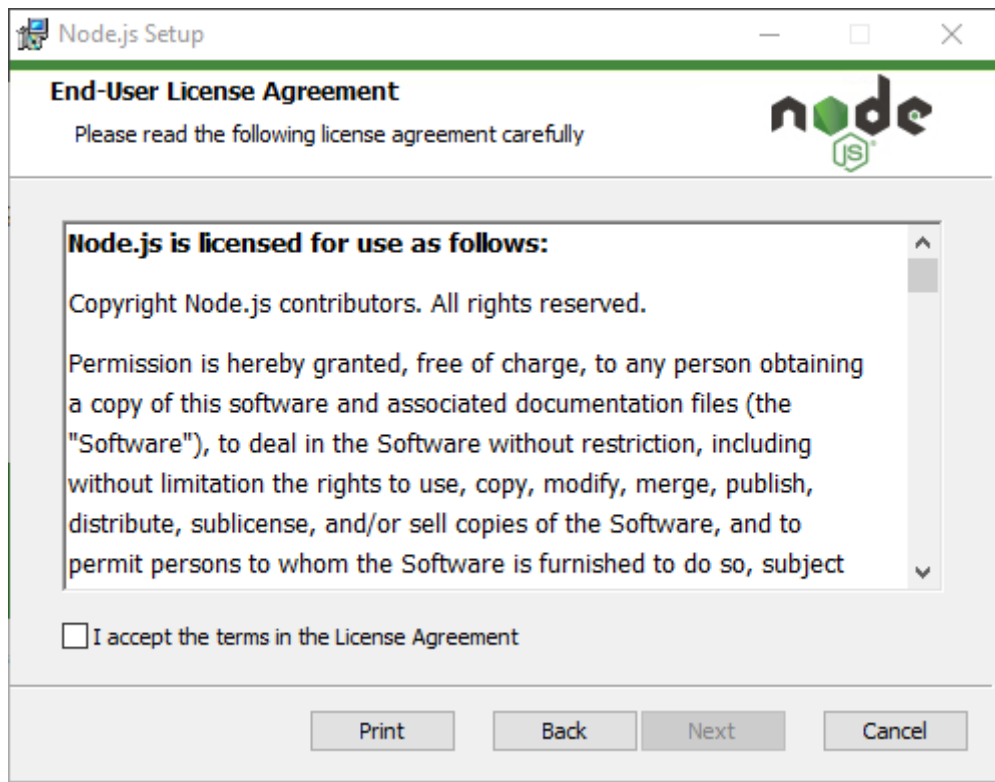
Otras Descargas | Cambios | Documentación del API

O eche un vistazo a la Programa de soporte a largo plazo (LTS)

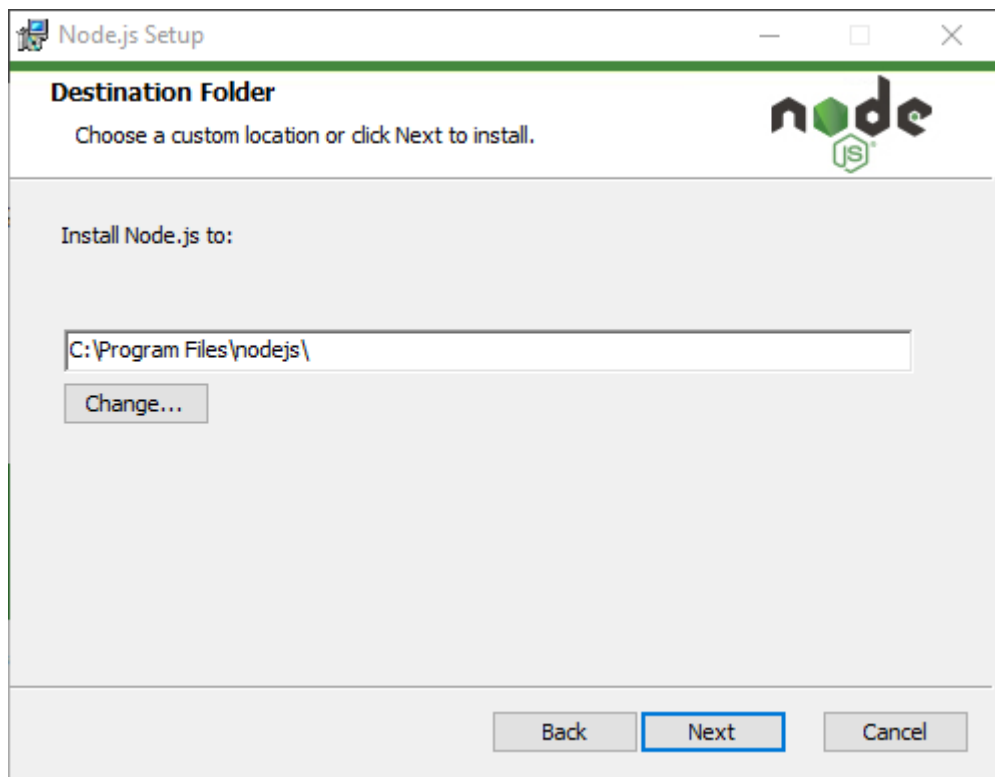
Y a continuación seguiremos los siguientes pasos para su instalación:



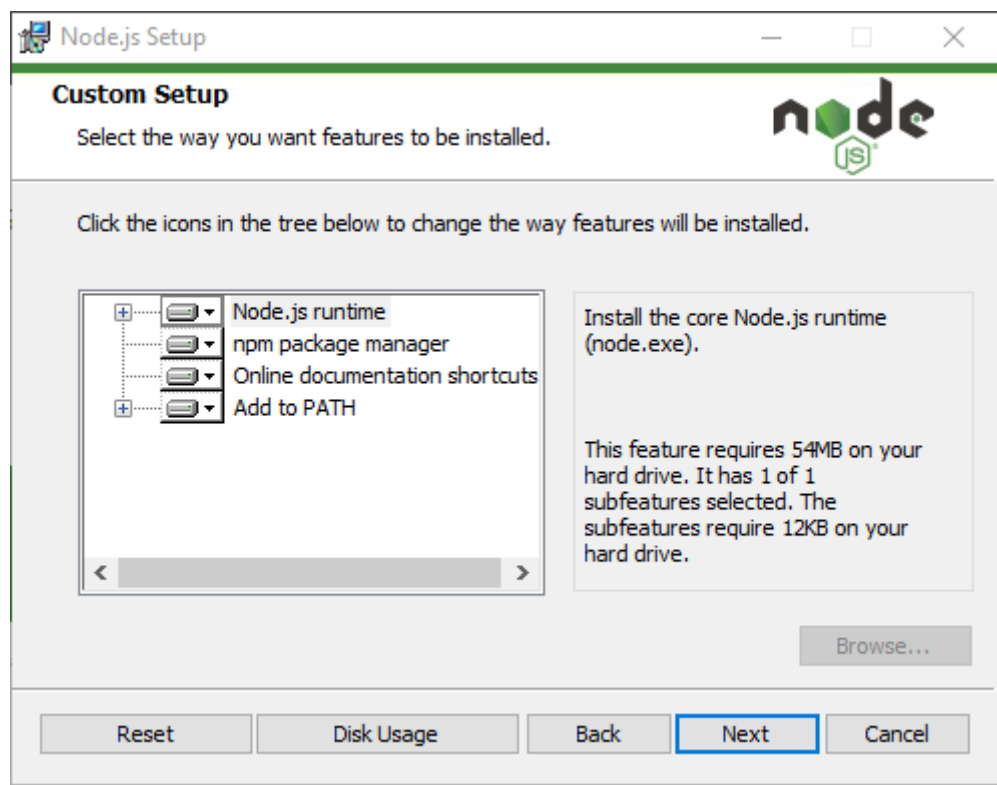
Aceptamos términos y condiciones:



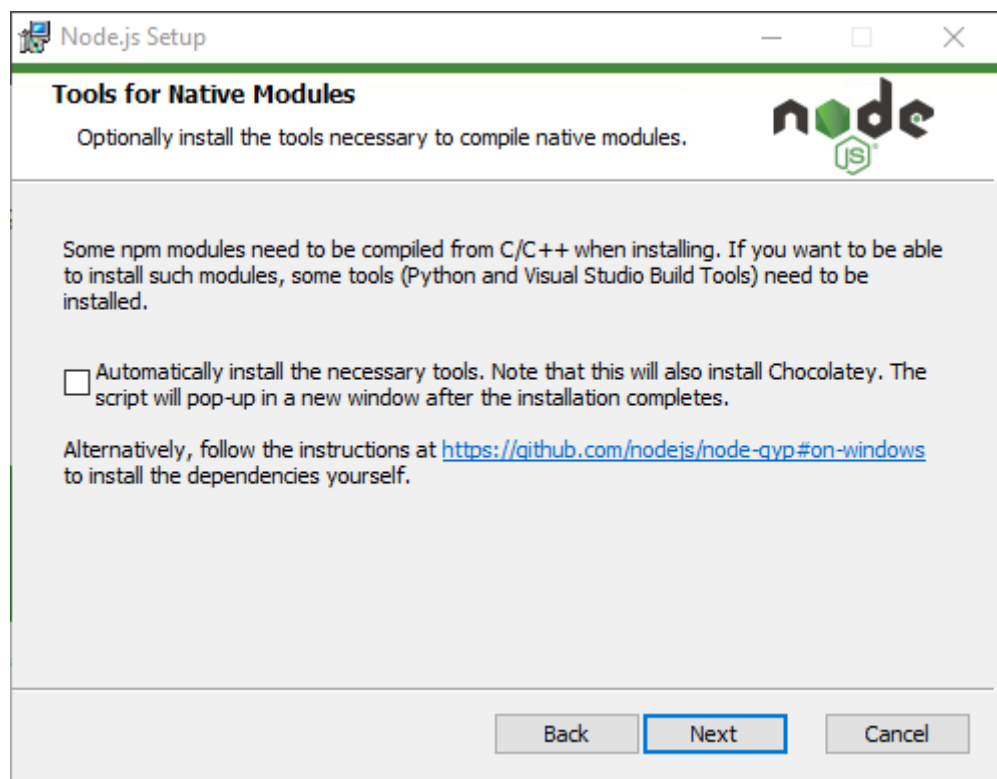
Indicamos el lugar de instalación:



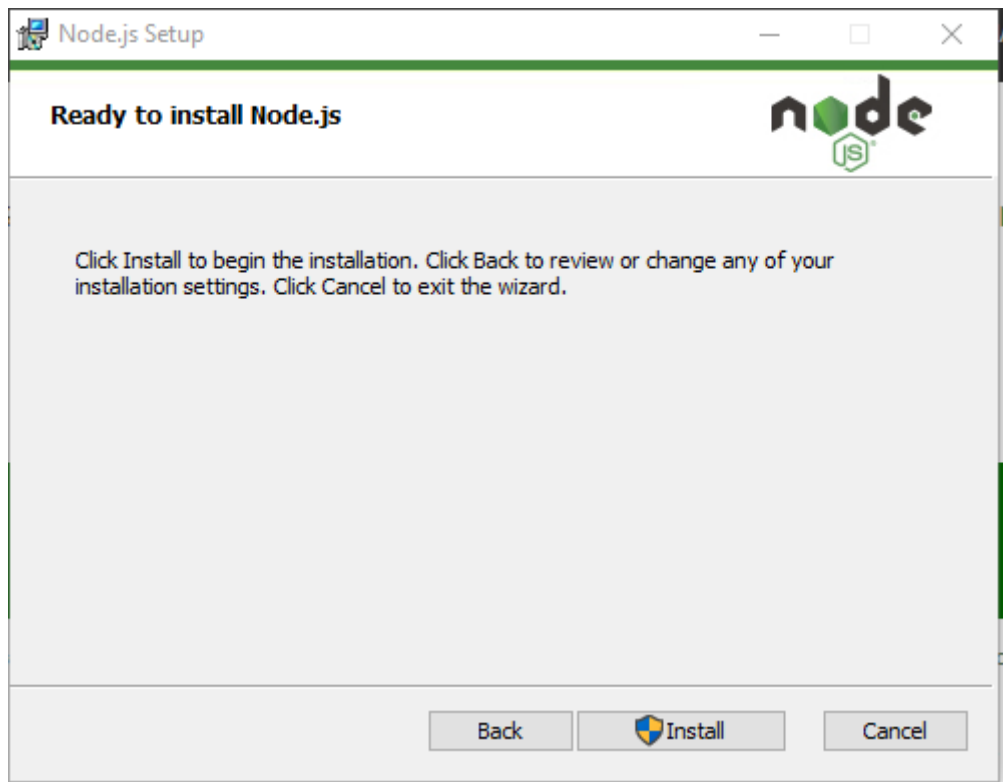
Pulsamos en NEXT:



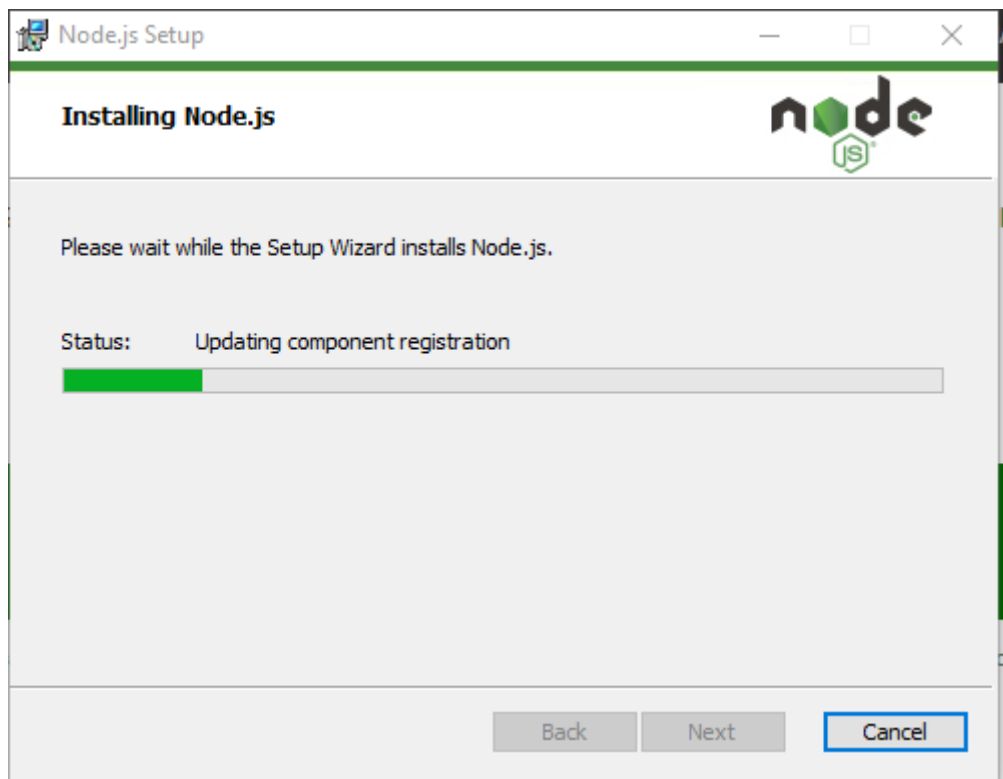
Pulsamos en NEXT y no activamos la casilla vacía:



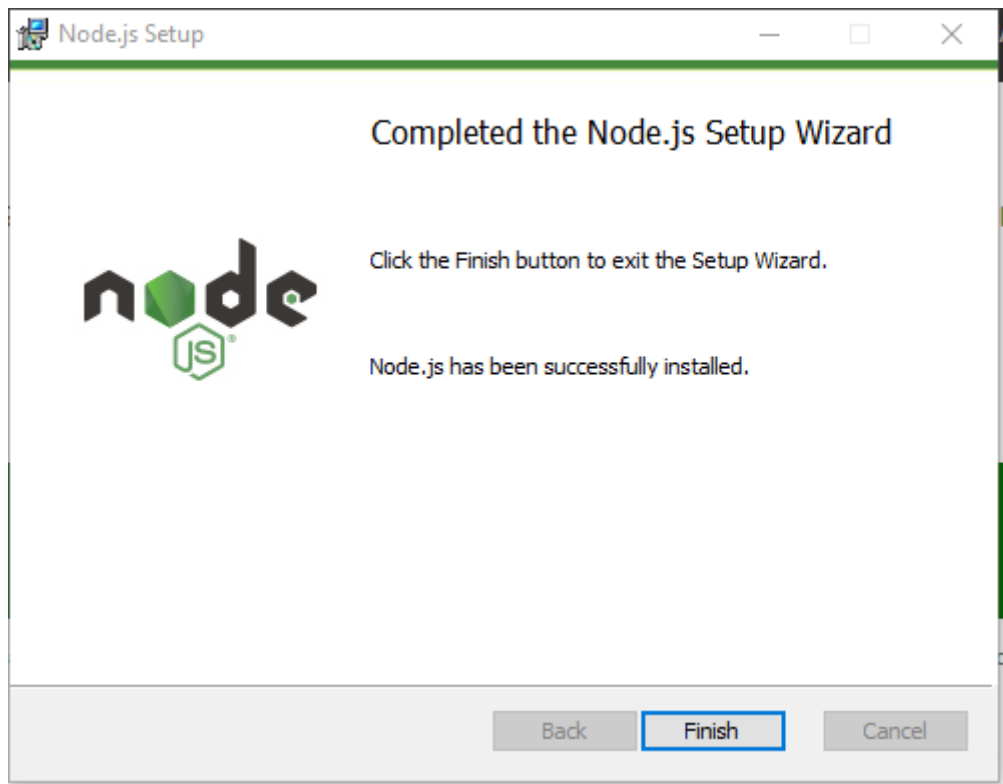
Pulsamos en Install:



Esperamos a la instalación:



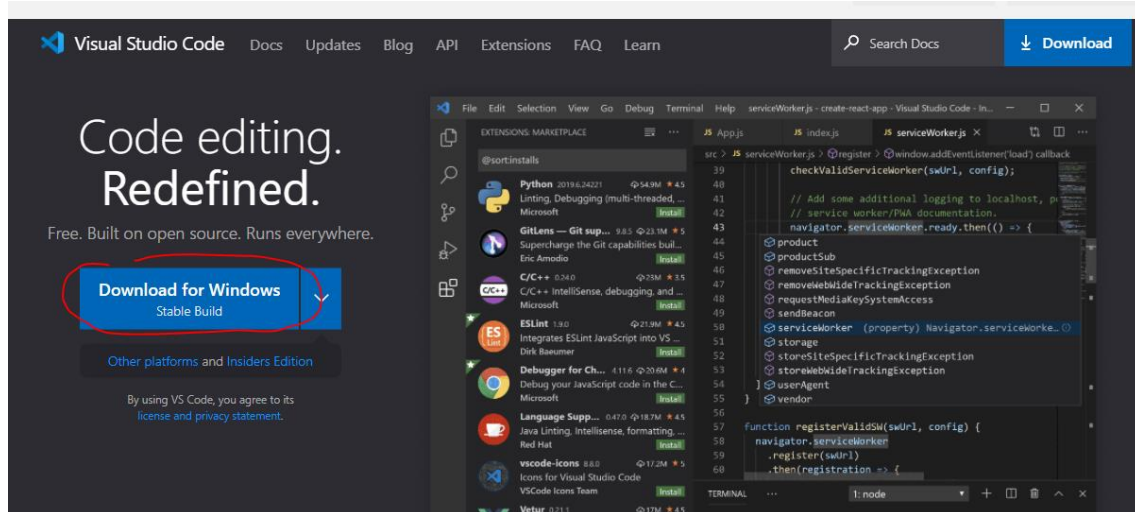
Finalizamos la instalación:



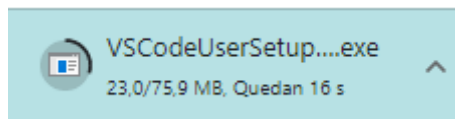
Instalación de visual Studio Code:

Para la instalación de visual Studio Code accederemos a la siguiente URL y pulsaremos en descargar para Windows:

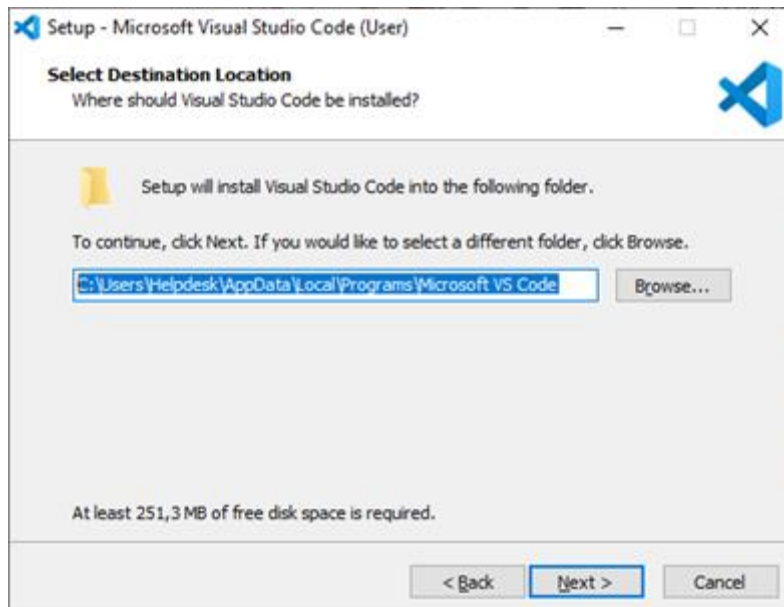
<https://code.visualstudio.com/>



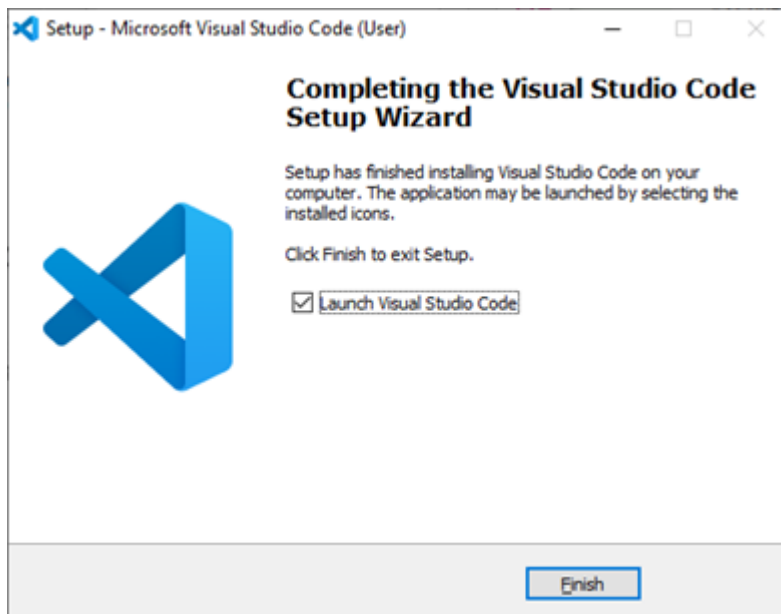
Esperamos a que termine la descarga y pulsamos en instalador:



Pulsamos next en todas las opciones:



Y esperamos a que se complete la instalación:



Instalación Global de TypeScript:

Globally Installing TypeScript

It can be handy to have TypeScript available across all projects, often by using project-wide installation over a global install so that they can benefit from updates.

via npm

You can use npm to install TypeScript globally, this means that you can use the `tsc` command anywhere in your terminal.


To do this, run `npm install -g typescript`. This will install the latest version (currently 4.4).

An alternative is to use `npmx` when you have to run `tsc` for one-off occasions.

Fuente: <https://www.typescriptlang.org/download>

Ahora procederemos a realizar una demostración de como se debería de instalar globalmente TypeScript:

Abrimos la terminal de powershell y escribimos la siguiente orden. MUY IMPORTANTE, debe de especificar la opción -g para instalar globalmente:

 Windows PowerShell

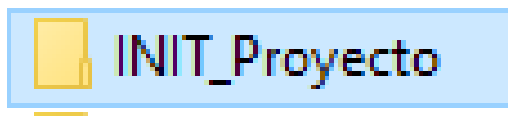
```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

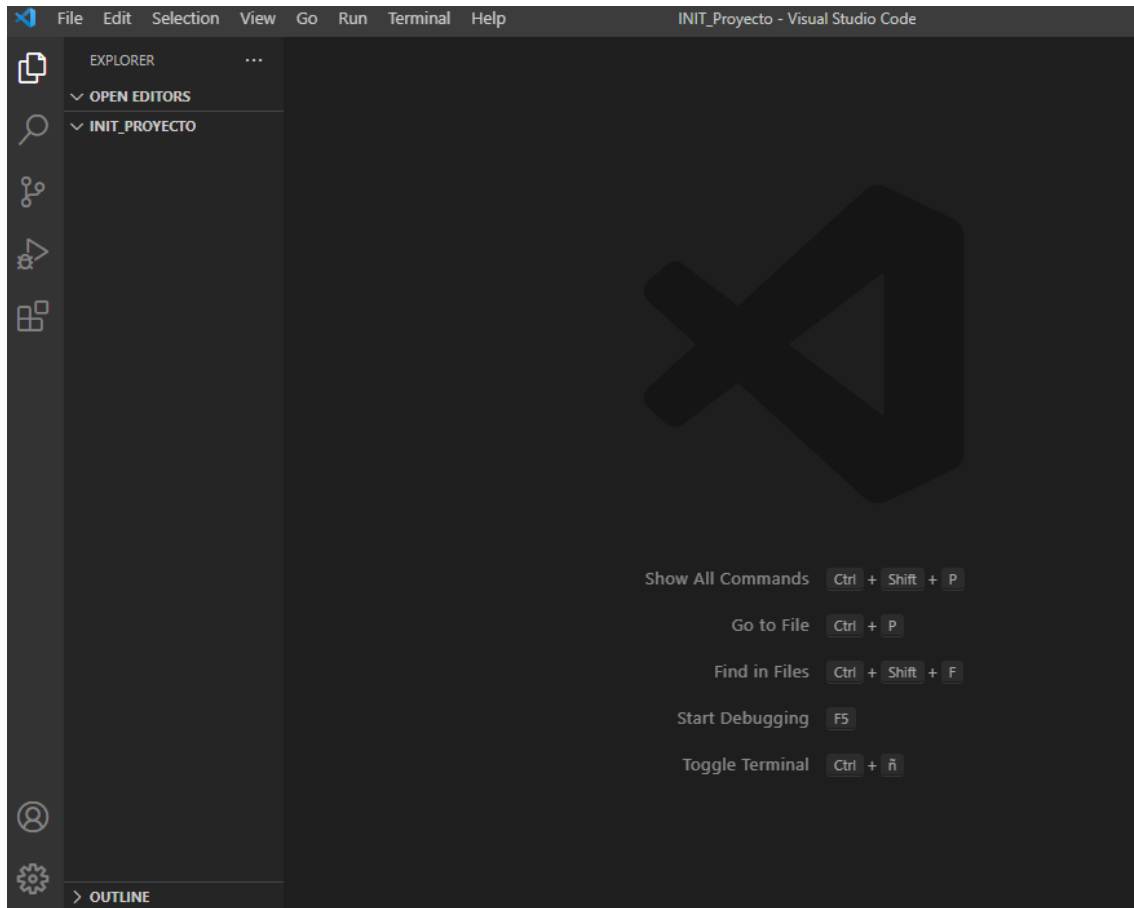
PS C:\Users\rafag> npm install -g typescript
```

Inicialización de un proyecto:

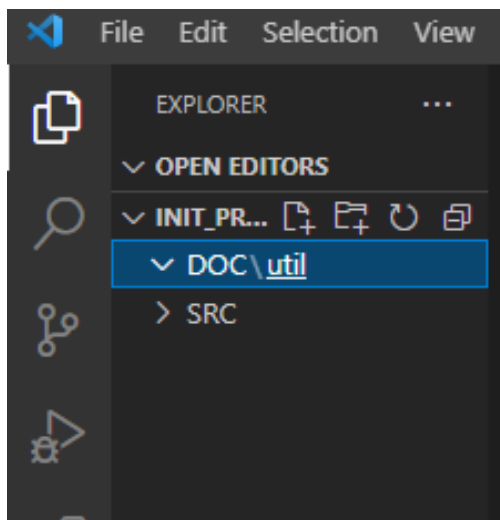
En primer lugar creamos una carpeta la cual alojara el proyecto:



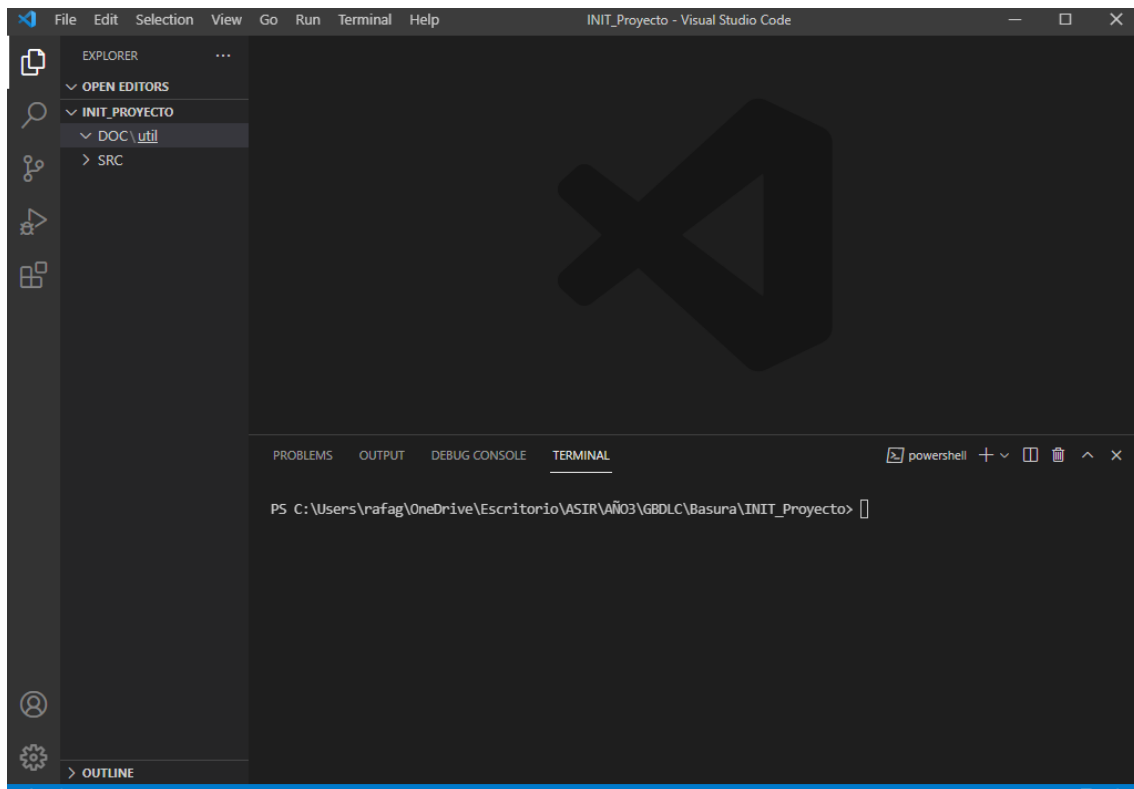
A continuación arrastramos a nuestro editor la carpeta:



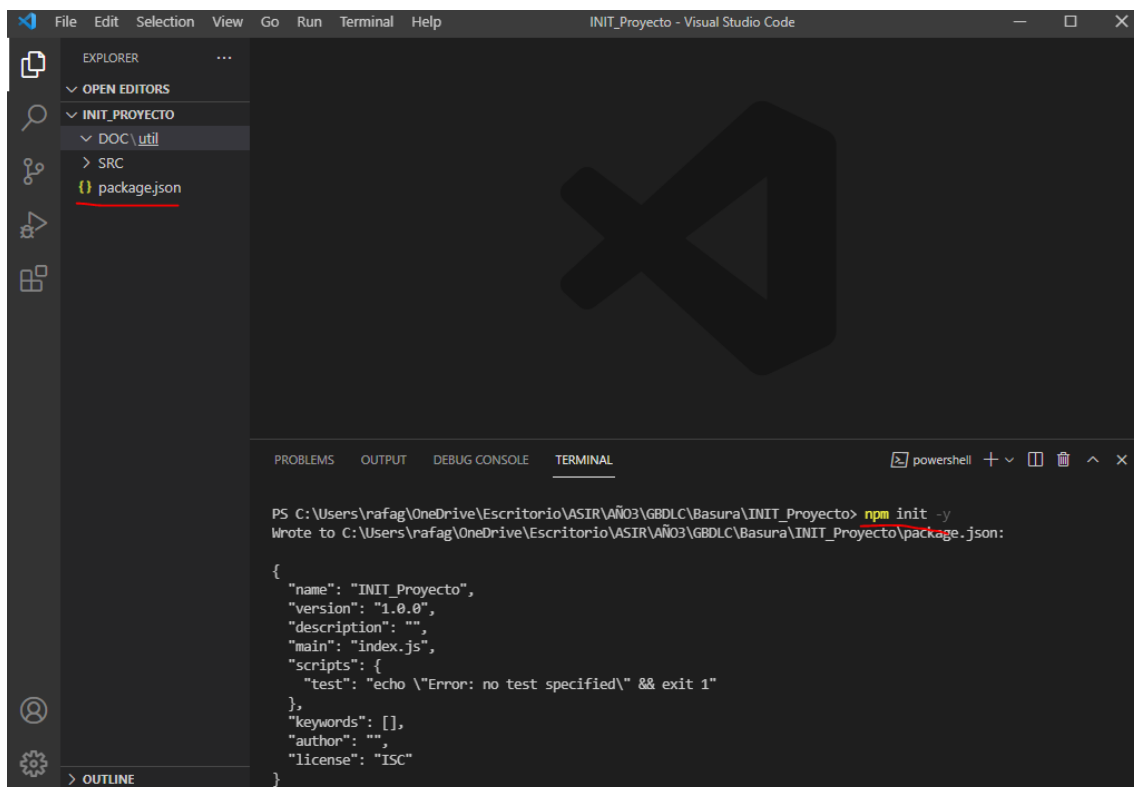
Procedemos a crear la estructura principal:



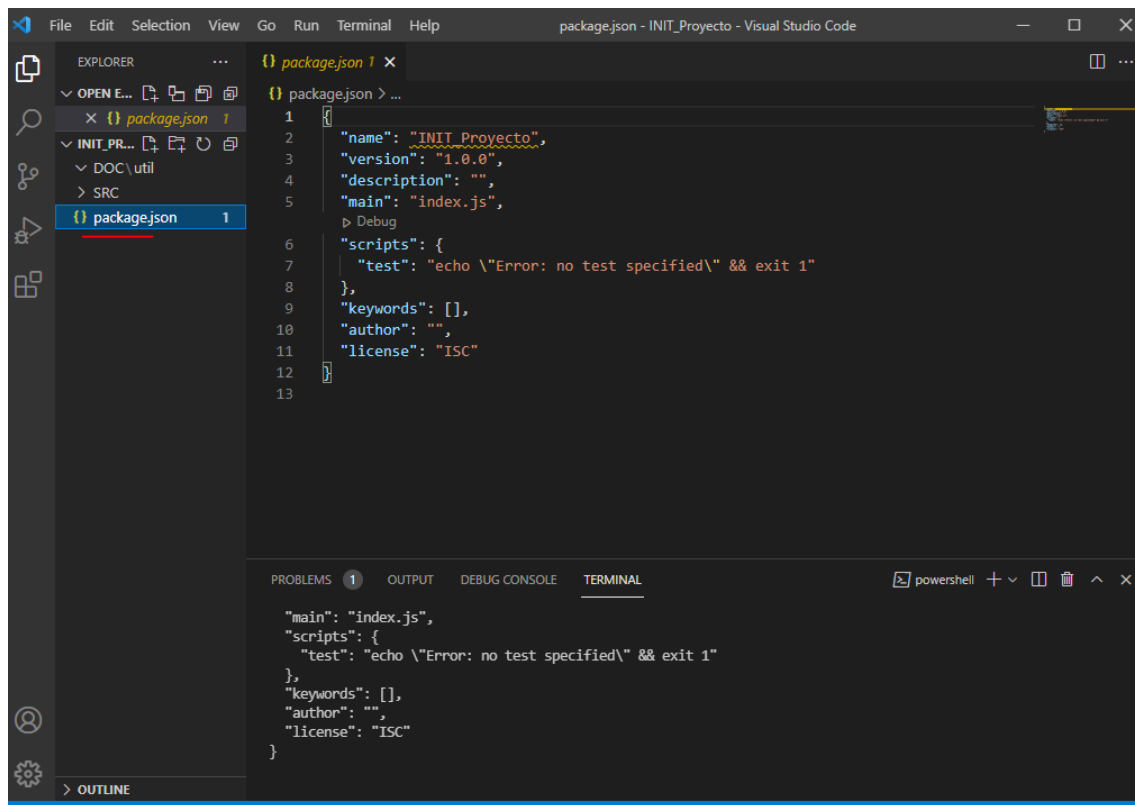
A continuación procedemos a inicializar el proyecto para ello abrimos el terminal:



Ejecutamos el comando `npm init -y`:



Entonces se creará el documento package.json:

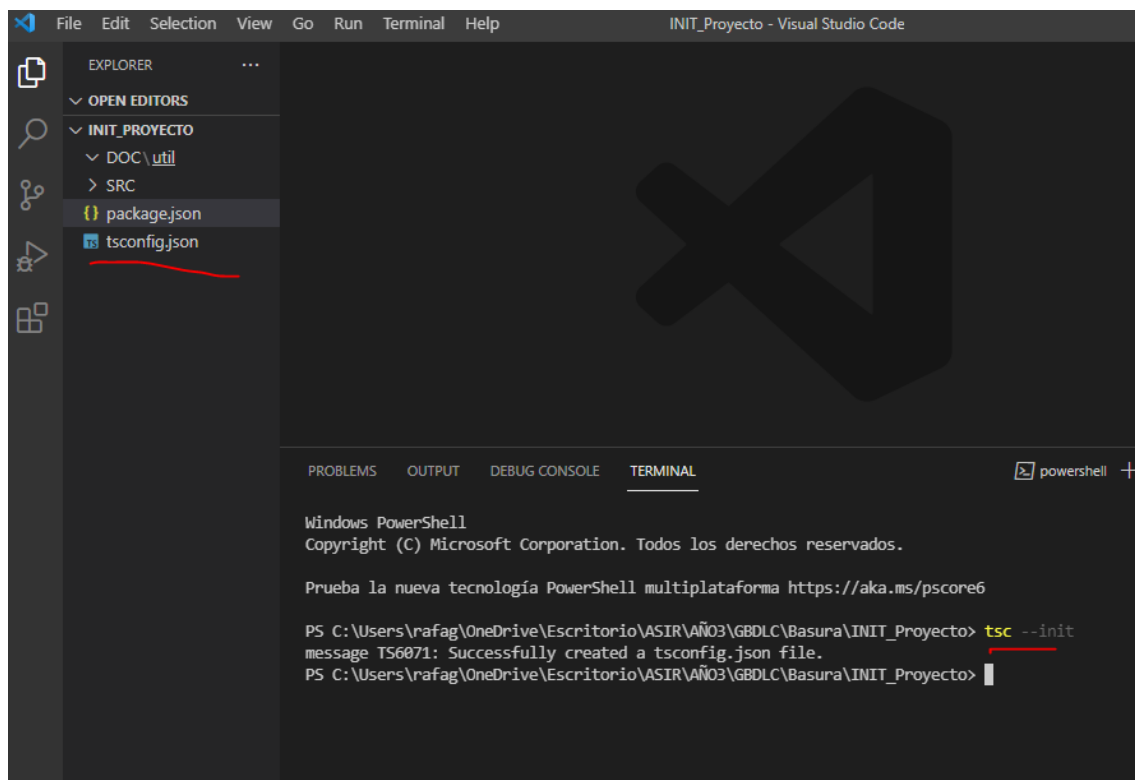


```
package.json - INIT_Proyecto - Visual Studio Code

package.json > ...
1 {
2   "name": "INIT_Proyecto",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \\\"Error: no test specified\\\" && exit 1"
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC"
12 }
13

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL
"main": "index.js",
"scripts": {
  "test": "echo \\\"Error: no test specified\\\" && exit 1"
},
"keywords": [],
"author": "",
"license": "ISC"
}
```

Ahora ejecutaremos `tsc --init`, lo cual creará el archivo `tsconfig.json`:



```
INIT_Proyecto - Visual Studio Code

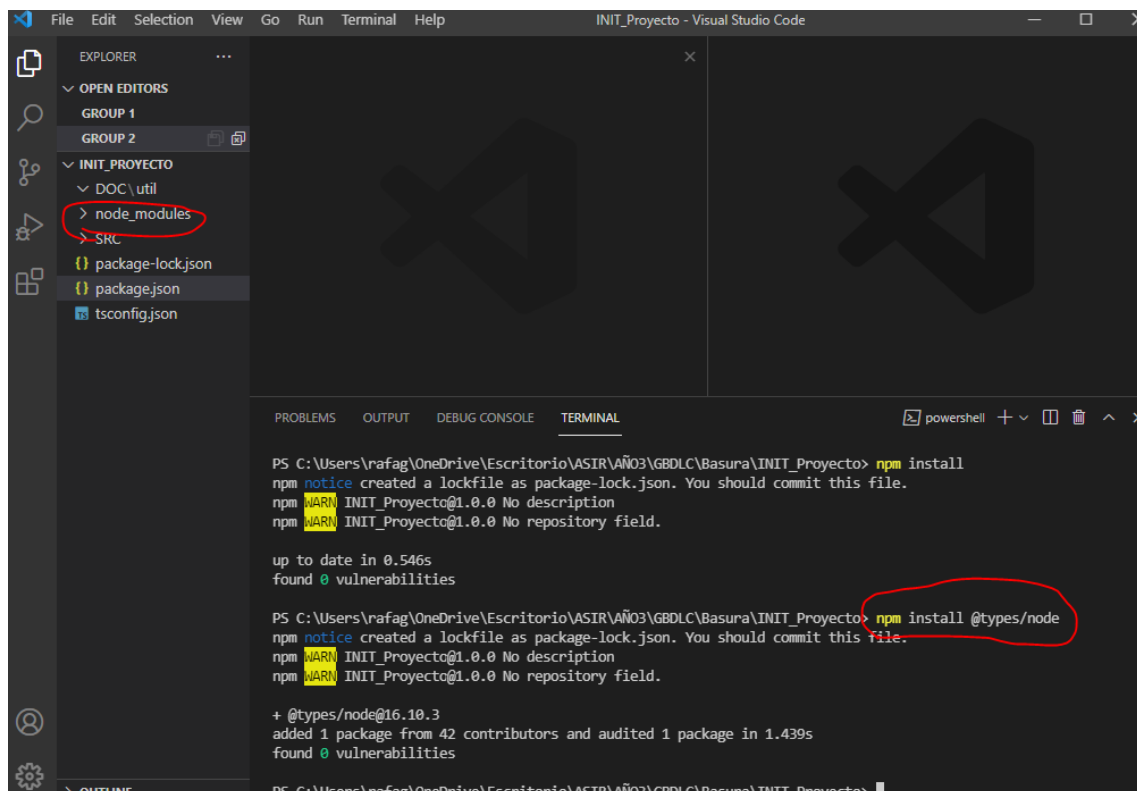
EXPLORER
OPEN EDITORS
INIT_PROYECTO
  DOC \ util
    > SRC
      package.json
      tsconfig.json

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\rafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\Basura\INIT_Proyecto> tsc --init
message TS6071: Successfully created a tsconfig.json file.
PS C:\Users\rafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\Basura\INIT_Proyecto>
```

Ahora ejecutaremos el siguiente comando, el cual creará el directorio “npm install @types/node”:



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left. The 'node_modules' directory is highlighted. The Terminal panel at the bottom shows the command 'npm install @types/node' being executed. The output indicates that the package was successfully installed, adding 1 package from 42 contributors and auditing 1 package in 1.439s, with 0 vulnerabilities found.

```
PS C:\Users\rafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\Basura\INIT_Proyecto> npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN INIT_Proyecto@1.0.0 No description
npm WARN INIT_Proyecto@1.0.0 No repository field.

up to date in 0.546s
found 0 vulnerabilities

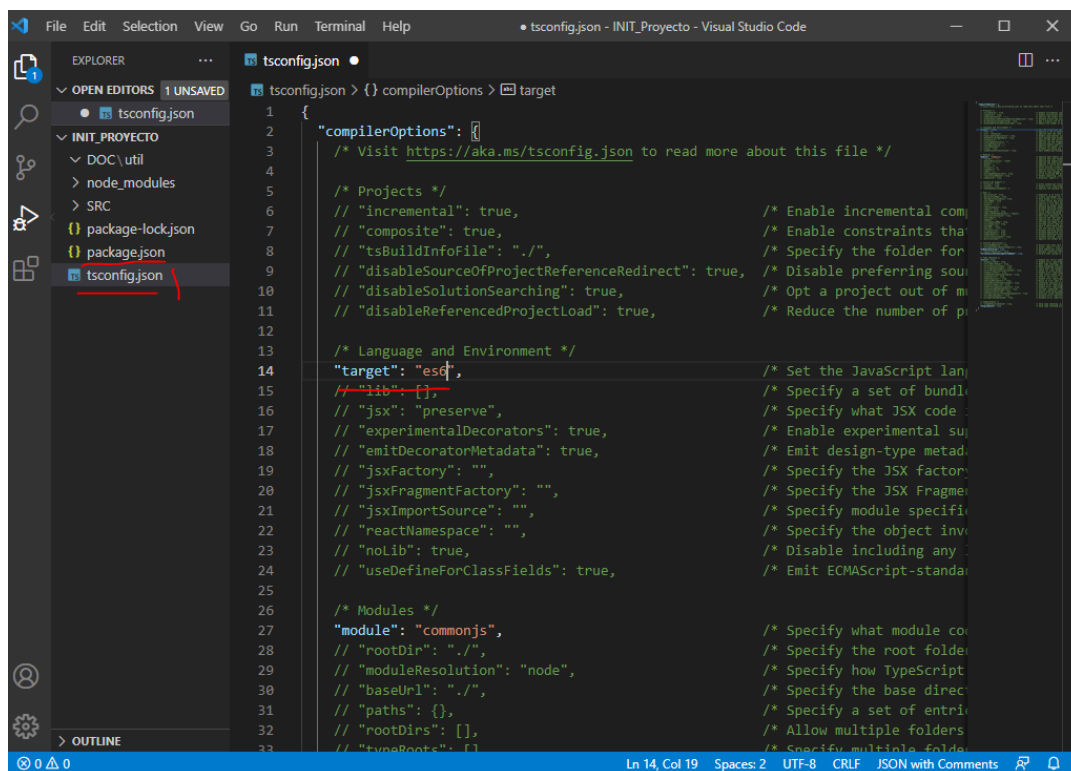
PS C:\Users\rafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\Basura\INIT_Proyecto> npm install @types/node
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN INIT_Proyecto@1.0.0 No description
npm WARN INIT_Proyecto@1.0.0 No repository field.

+ @types/node@16.10.3
added 1 package from 42 contributors and audited 1 package in 1.439s
found 0 vulnerabilities

PS C:\Users\rafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\Basura\INIT_Proyecto>
```

Ahora procederemos a mostrar diferentes modificaciones que tendremos que hacer en el archivo tsconfig.json:

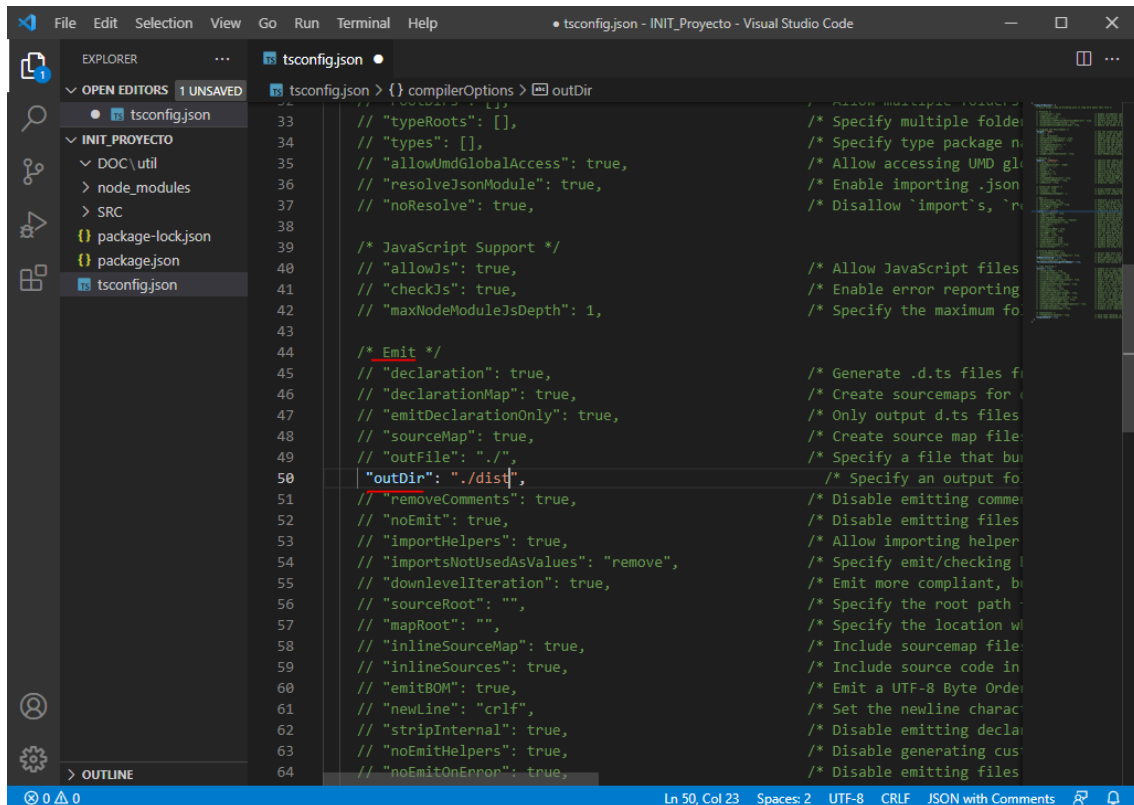
En primer lugar en la línea target indicaremos como mínimo la versión es6:



The screenshot shows the Visual Studio Code interface with the tsconfig.json file open in the editor. The 'target' property is highlighted, and its value is 'es6'. The Explorer sidebar on the left shows the file structure, and the Outline panel on the right shows the file's structure.

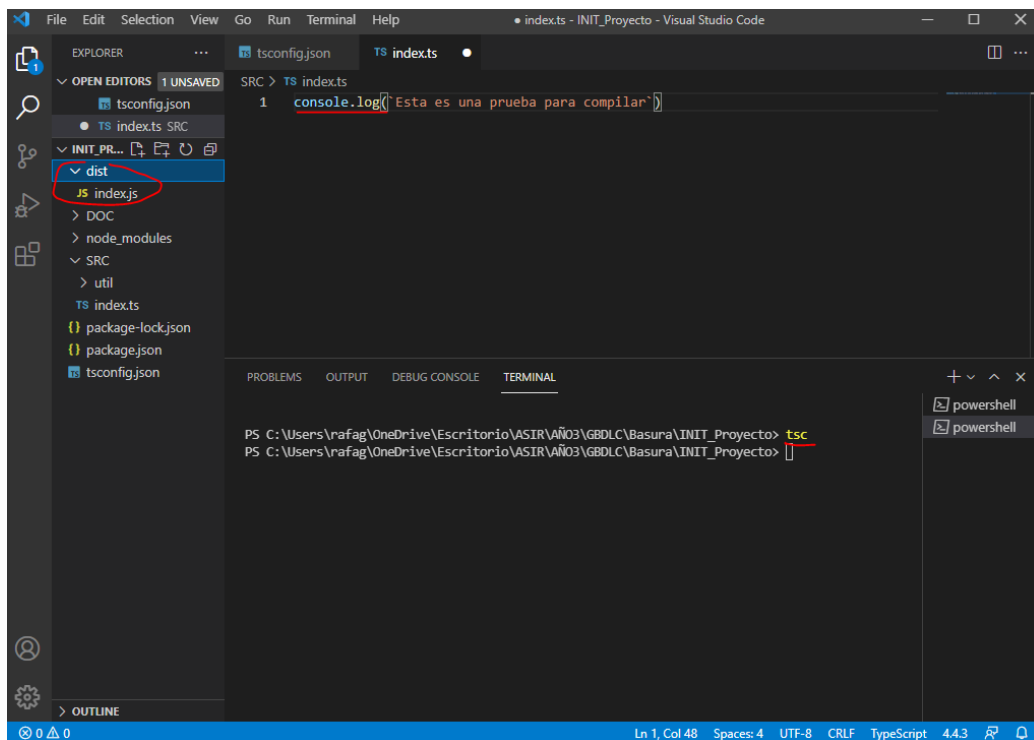
```
1 {
2   "compilerOptions": {
3     /* Visit https://aka.ms/tsconfig.json to read more about this file */
4
5     /* Projects */
6     // "incremental": true,           /* Enable incremental compilation */
7     // "composite": true,             /* Composite projects to enable faster compilation by reusing previous compilation results */
8     // "tsBuildInfoFile": "./",       /* Specify the folder for .tsbuildinfo files, which allows for faster compilation times when using incremental compilation */
9     // "disableSourceOfProjectReferenceRedirect": true, /* Disable preferring source files instead of declaration files when resolving project references */
10    // "disableSolutionSearching": true, /* Opt a project out of the automatic search for project references */
11    // "disableReferencedProjectLoad": true, /* Reduce the number of projects loaded automatically byVS Code */
12
13    /* Language and Environment */
14    "target": "es6",                 /* Set the JavaScript language version for emitting. Defaults to 'es5'. */
15    // "lib": [],                     /* Specify a set of bundled library declaration files that describe the target runtime environment. */
16    // "jsx": "preserve",              /* Specify what JSX code is generated. */
17    // "experimentalDecorators": true, /* Enable experimental support for decorators. */
18    // "emitDecoratorMetadata": true,  /* Emit design-type metadata for decorated declarations in source files. */
19    // "jsxFactory": "",                /* Specify the JSX factory function used when transforming JSX. Defaults to 'React.createElement' or 'h' for 'react/jsx-runtime'. */
20    // "jsxFragmentFactory": "",        /* Specify the JSX Fragment Factory used when transforming JSX. Defaults to 'React.Fragment' or 'Fragment' for 'react/jsx-runtime'. */
21    // "jsxImportSource": "",           /* Specify module specifier used to import the JSX factory functions. */
22    // "reactNamespace": "",            /* Specify the object used for naming React and ReactDOM. */
23    // "noLib": true,                  /* Disable including any library files. */
24    // "useDefineForClassFields": true, /* Emit ECMAScript-standard-compliant class fields. */
25
26    /* Modules */
27    "module": "commonjs",            /* Specify what module code is generated. */
28    // "rootDir": "./",               /* Specify the root folder for the current project, and the rootDir of the project's source files. */
29    // "moduleResolution": "node",     /* Specify how TypeScript looks up a file from a given module specifier. */
30    // "baseUrl": "./",               /* Specify the base directory to resolve non-relative module names. */
31    // "paths": {},                   /* Specify a set of entries that re-map imports to additional lookup locations. */
32    // "rootDirs": [],                /* Allow multiple folders to be treated as one when resolving modules. */
33    // "typeRoots": [],               /* Specify multiple folders that act as `node_modules` or `node_modules/@types`. */
```

En segundo lugar tendremos que ir a la línea que dice “Emit” e indicar que el código compilado se añadirá a la carpeta /dist:



```
tsconfig.json
33 // "typeRoots": [],
34 // "types": [],
35 // "allowUmdGlobalAccess": true,
36 // "resolveJsonModule": true,
37 // "noResolve": true,
38
39 /* JavaScript Support */
40 // "allowJs": true,
41 // "checkJs": true,
42 // "maxNodeModuleJsDepth": 1,
43
44 /* Emit */
45 // "declaration": true,
46 // "declarationMap": true,
47 // "emitDeclarationOnly": true,
48 // "sourceMap": true,
49 // "outFile": "./",
50 "outDir": "./dist",
51 // "removeComments": true,
52 // "noEmit": true,
53 // "importHelpers": true,
54 // "importsNotUsedAsValues": "remove",
55 // "downlevelIteration": true,
56 // "sourceRoot": "",
57 // "mapRoot": "",
58 // "inlineSourceMap": true,
59 // "inlineSources": true,
60 // "emitBOM": true,
61 // "newline": "crlf",
62 // "stripInternal": true,
63 // "noEmitHelpers": true,
64 // "noEmitOnError": true,
```

A continuación, añadiremos algo de código en el archivo index.ts con el fin de ejecutar el comando tsc y compilar los archivos. Como se puede observar una vez lanzado el comando tsc nos genera el directorio dist en el cual se aloja index.js el cual sería el resultado de la compilación del index.ts:



```
index.ts
1 console.log("Esta es una prueba para compilar")
```

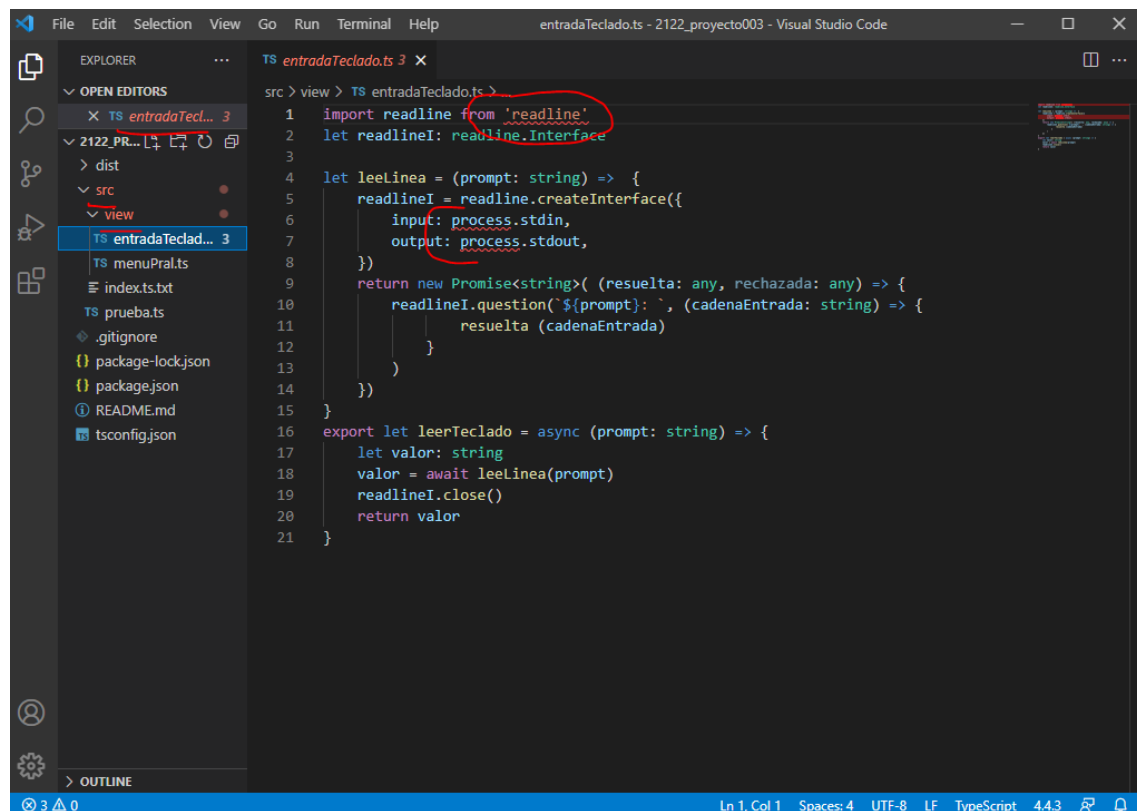
```
PS C:\Users\rafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\Basura\INIT_Proyecto> tsc
PS C:\Users\rafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\Basura\INIT_Proyecto>
```

Instalación de un proyecto ya existente:

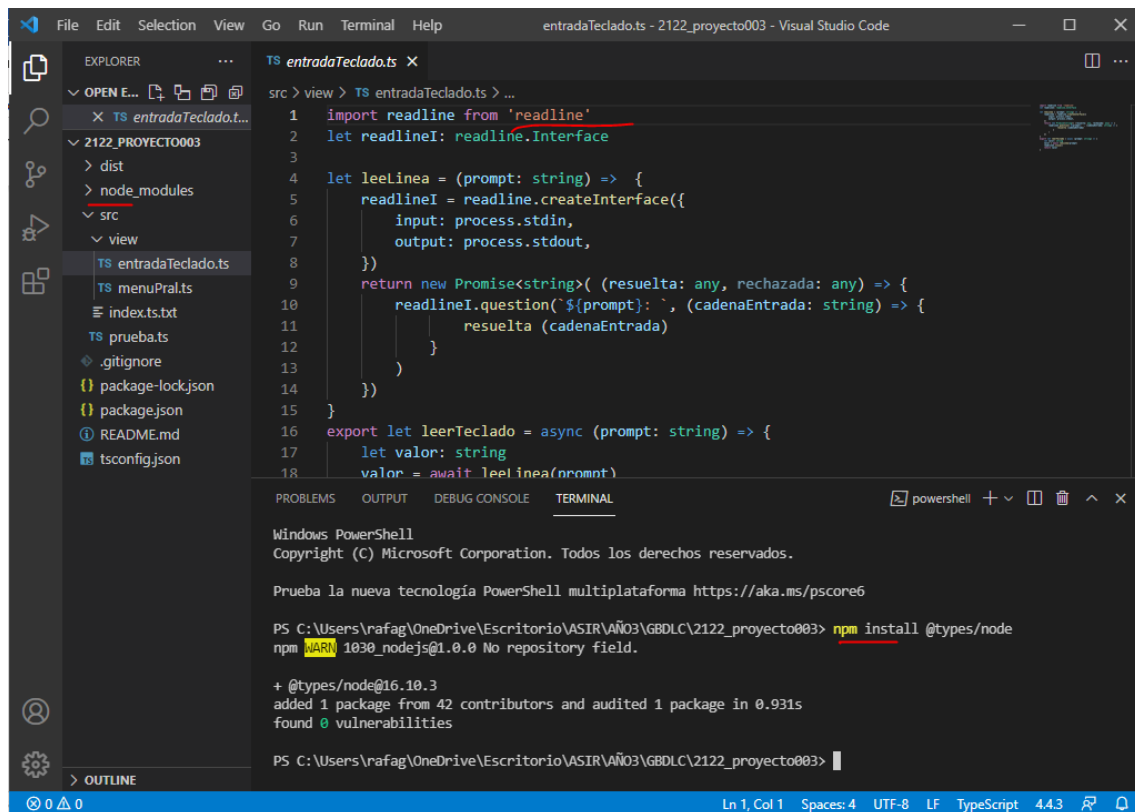
Para la instalación de un proyecto ya existente utilizaremos el proyecto003 el cual hemos descargado desde la plataforma Moodle:



Una vez que lo tenemos en nuestro editor podemos observar que la carpeta node_modules no existe, además de que el editor detecta varios errores, para solucionarlo empezaremos por ejecutar el comando `npm install @types/node`:



Veremos cual es el resultado después de ejecutar `npm install @types/node`:



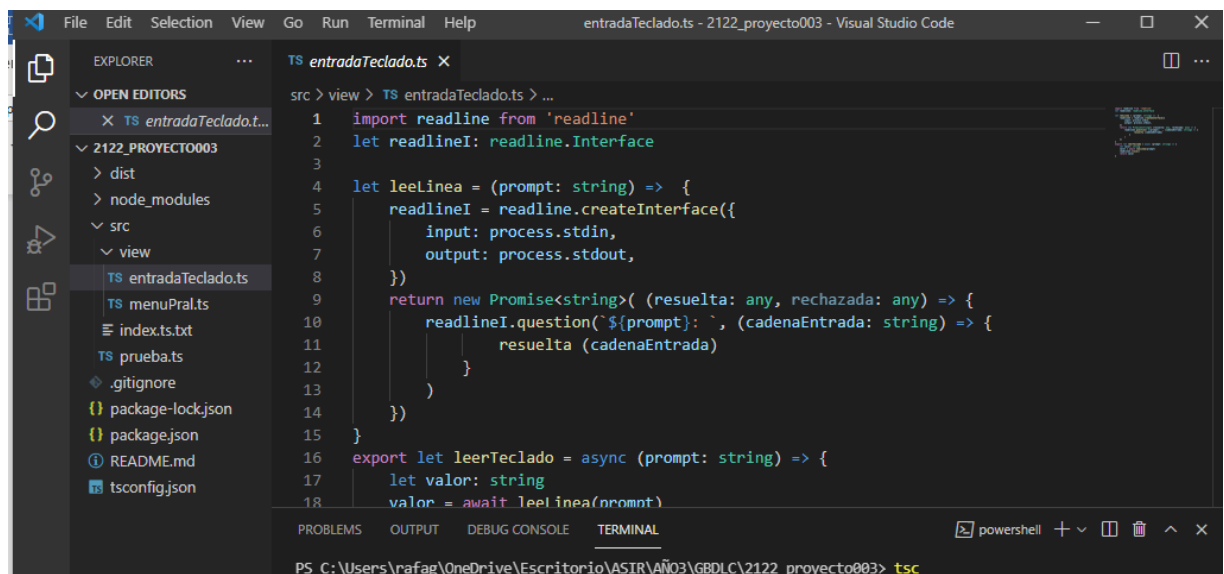
The screenshot shows the Visual Studio Code interface. The Explorer panel on the left displays the project structure for '2122_PROYECTO003', including a 'node_modules' directory. The main editor shows the file 'entradaTeclado.ts' with the following TypeScript code:

```
1 import readline from 'readline'
2 let readlineI: readline.Interface
3
4 let leeLinea = (prompt: string) => {
5     readlineI = readline.createInterface({
6         input: process.stdin,
7         output: process.stdout,
8     })
9     return new Promise<string>((resuelta: any, rechazada: any) => {
10         readlineI.question(`${prompt}: `, (cadenaEntrada: string) => {
11             resuelta(cadenaEntrada)
12         })
13     })
14 }
15
16 export let leerTeclado = async (prompt: string) => {
17     let valor: string
18     valor = await leeLinea(prompt)
```

The terminal window at the bottom shows the output of the command `npm install @types/node` in a PowerShell session. The output indicates that the package was successfully installed, with a warning about the repository field and a note about the number of contributors and vulnerabilities audited.

Como podemos ver la carpeta `node_modules` ya esta disponible y los errores han desaparecido.

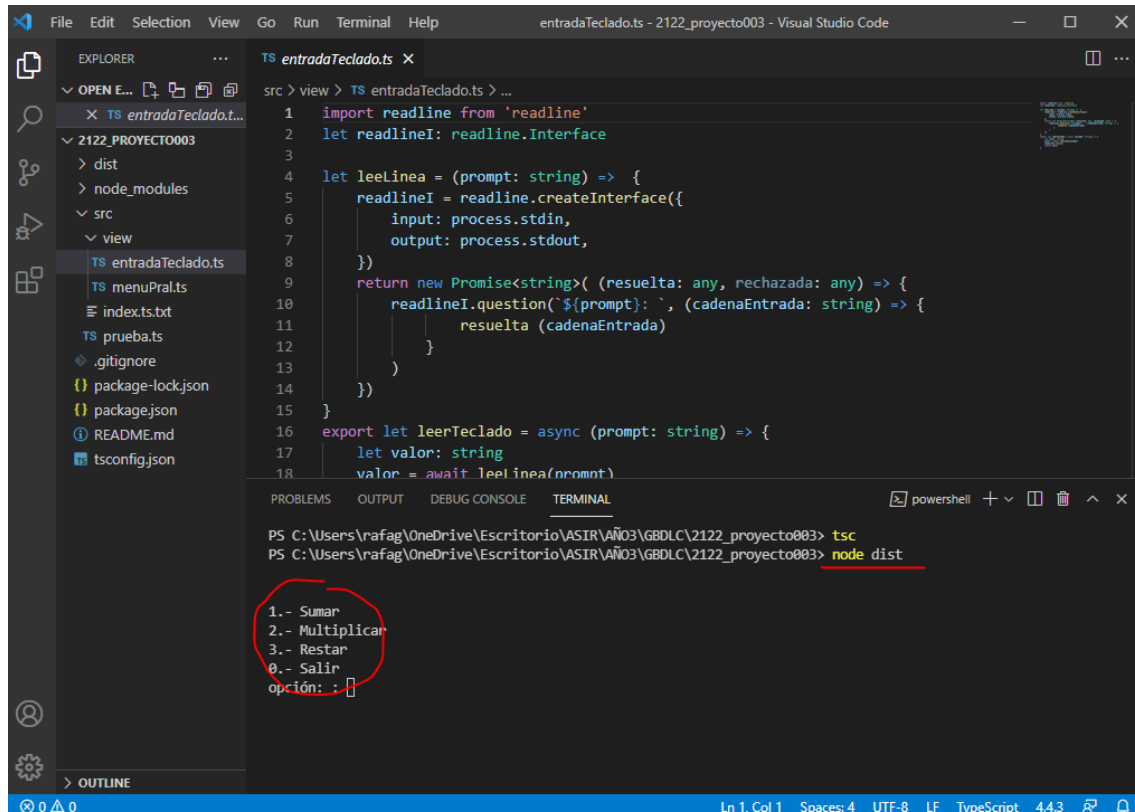
Ahora procederemos a compilar para comprobar si el proyecto funciona correctamente:



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left displays the project structure for '2122_PROYECTO003', including a 'node_modules' directory. The main editor shows the file 'entradaTeclado.ts' with the same TypeScript code as in the previous screenshot. The terminal window at the bottom shows the command `tsc` entered, which is used to compile the TypeScript code.

Y en último lugar ejecutamos para comprobar si todo está correcto:

Como podemos ver el programa funciona correctamente:



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows the project structure for '2122_PROYECTO003', with 'src' containing 'entradaTeclado.ts'. The main editor displays the code for 'entradaTeclado.ts', which uses the 'readline' module to create a CLI interface. The code defines a 'leeLinea' function that prompts the user and returns a Promise, and an 'leerTeclado' function that uses 'leeLinea' to read input. The terminal at the bottom shows the command 'tsc' being run to compile the TypeScript file, followed by 'node dist' to execute the compiled JavaScript. The output of the program is displayed in the terminal, showing a list of options: '1.- Sumar', '2.- Multiplicar', '3.- Restar', and '0.- Salir', followed by a prompt 'opción: ' where the user has entered a space character.

```
1 import readline from 'readline'
2 let readlineI: readline.Interface
3
4 let leeLinea = (prompt: string) => {
5   readlineI = readline.createInterface({
6     input: process.stdin,
7     output: process.stdout,
8   })
9   return new Promise<string>((resuelta: any, rechazada: any) => {
10     readlineI.question(`${prompt}: `, (cadenaEntrada: string) => {
11       resuelta(cadenaEntrada)
12     })
13   })
14 }
15
16 export let leerTeclado = async (prompt: string) => {
17   let valor: string
18   valor = await leeLinea(prompt)
19 }
```

```
PS C:\Users\nafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\2122_proyecto003> tsc
PS C:\Users\nafag\OneDrive\Escritorio\ASIR\AÑO3\GBDLC\2122_proyecto003> node dist

1.- Sumar
2.- Multiplicar
3.- Restar
0.- Salir
opción:  
```