Nuestro primer test unitario con JavaScript

DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Índice

1 Configurar el Framework

Framework para JavaScript

Para JavaScript vamos a utilizar el framework **Jest**. Si queremos configurarlo, debemos instalar:

1. Un IDE (el recomendado es **Visual Studio Code** (https://code.visualstudio.com)

2. **Node.js** (https://nodejs.org).

   Debemos tener en cuenta que si el código brindado no posee el archivo **package.json** debemos crearlo, ya que aquí se van a guardar todas las configuraciones de nuestro proyecto. Para crearlo debemos correr en la terminal el comando: **npm init -y**

3. JEST (https://jestjs.io/)

Para incluir Jest dentro de nuestro proyecto debemos correr en la terminal el comando: **npm install --save-dev jest** (tener en cuenta que esto debemos realizarlo en cada uno de los proyectos que vamos a trabajar).

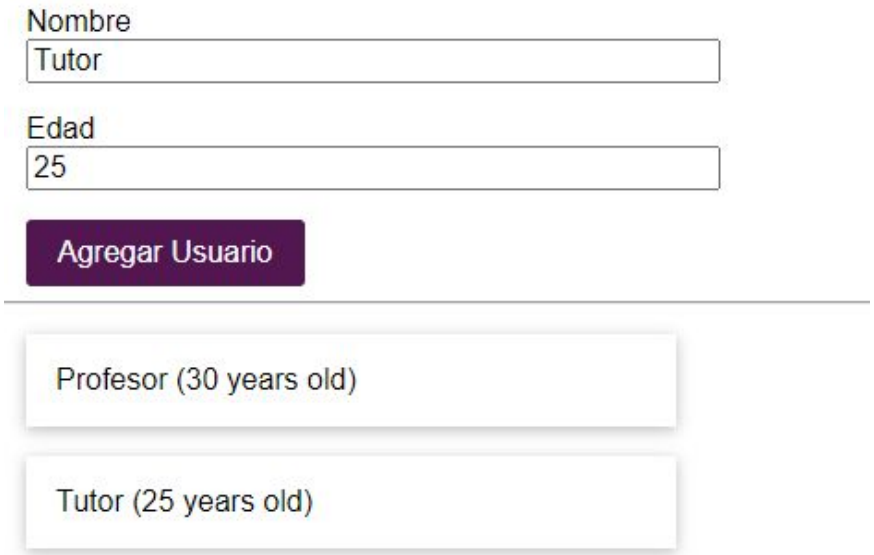
2 Nuestro primer test

Crear nuestro primer unit test

1. Bajar el código fuente del siguiente repositorio:

   https://github.com/academind/js-testing-introduction/tree/starting-setup.

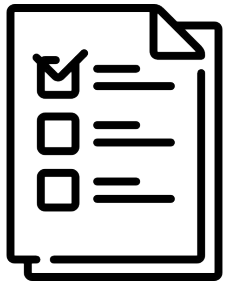
Este programa solicita el ingreso de **Nombre** y **Edad** y al presionar el botón **Agregar Usuario** arma una lista con los datos ingresados.



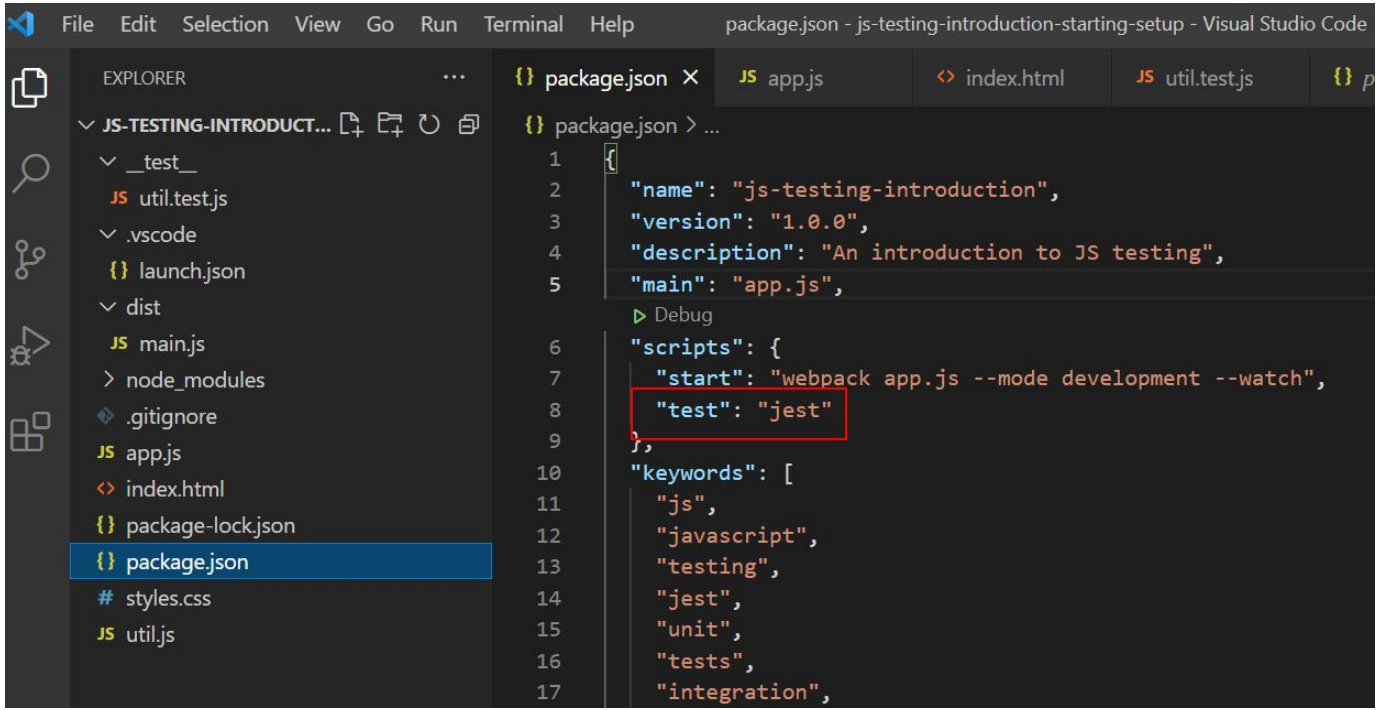
2

Configurar Jest como ejecutor de las pruebas:

1. Ir al archivo **package.json**.
2. En la parte de scripts/test debemos reemplazar **“echo “Error: no test specified” && exit 1”** por **“jest”**. De esta forma indicamos que vamos a correr test con nuestro framework Jest.



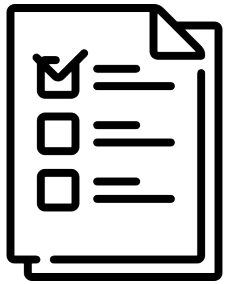
2


File Edit Selection View Go Run Terminal Help
package.json - js-testing-introduction-starting-setup - Visual Studio Code
EXPLORER
JS-TESTING-INTRODUCT...
__test__
util.test.js
.vscode
launch.json
dist
main.js
node_modules
.gitignore
app.js
index.html
package-lock.json
package.json
styles.css
util.js
package.json
app.js
index.html
util.test.js
package.json > ...
{
"name": "js-testing-introduction",
"version": "1.0.0",
"description": "An introduction to JS testing",
"main": "app.js",
Debug
"scripts": {
"start": "webpack app.js --mode development --watch",
"test": "jest"
},
"keywords": [
"js",
"javascript",
"testing",
"jest",
"unit",
"tests",
"integration",

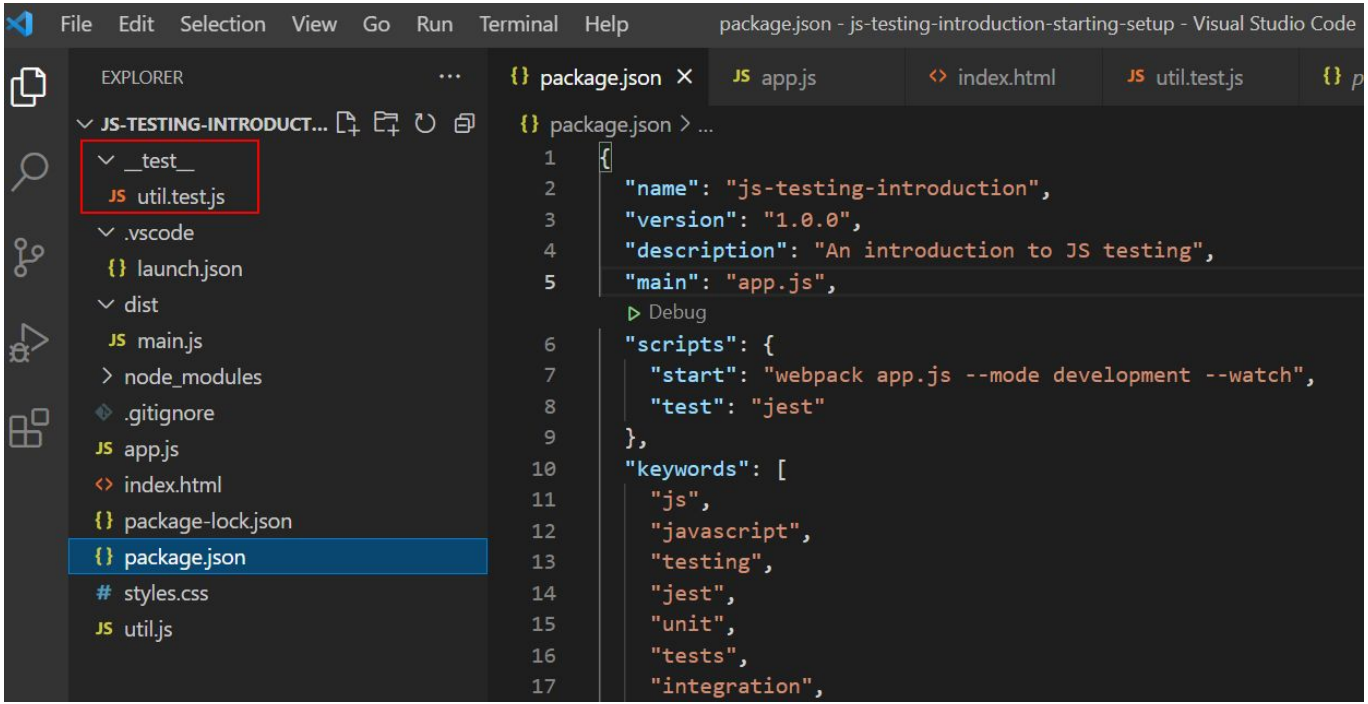
3

Crear la carpeta y archivo de prueba:

1. Crear la carpeta **__test__** donde se encontrarán todos los archivos de prueba.
2. Crear el archivo de prueba ***NombreArchivo*.test.js**. Para que Jest reconozca los archivos de prueba deben terminar en **.test.js**. En este caso nuestro archivo se llamará **util.test.js**.

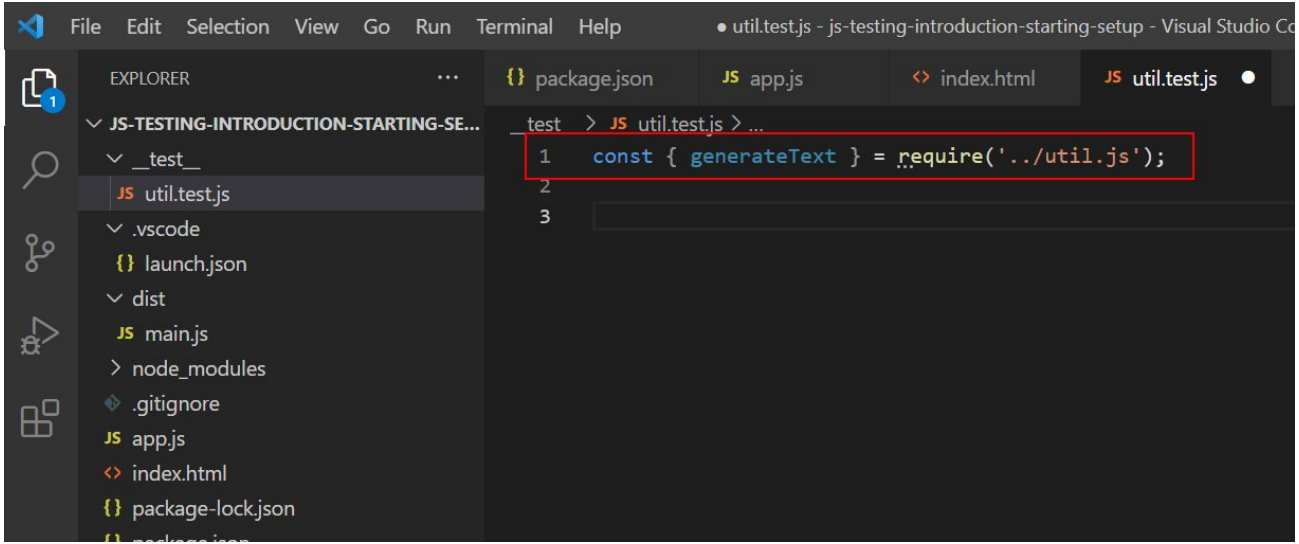


3


File Edit Selection View Go Run Terminal Help
package.json - js-testing-introduction-starting-setup - Visual Studio Code
EXPLORER
JS-TESTING-INTRODUCT...
__test__
util.test.js
.vscode
launch.json
dist
main.js
node_modules
.gitignore
app.js
index.html
package-lock.json
package.json
styles.css
util.js
package.json
app.js
index.html
util.test.js
package.json > ...
{
"name": "js-testing-introduction",
"version": "1.0.0",
"description": "An introduction to JS testing",
"main": "app.js",
Debug
"scripts": {
"start": "webpack app.js --mode development --watch",
"test": "jest"
},
"keywords": [
"js",
"javascript",
"testing",
"jest",
"unit",
"tests",
"integration",

4

Importar el código a probar: debemos importar la sección del código que queremos probar. En este caso vamos a probar el que se encuentra en el archivo **util.js**, ya que ahí se encuentra la lógica de nuestro programa.



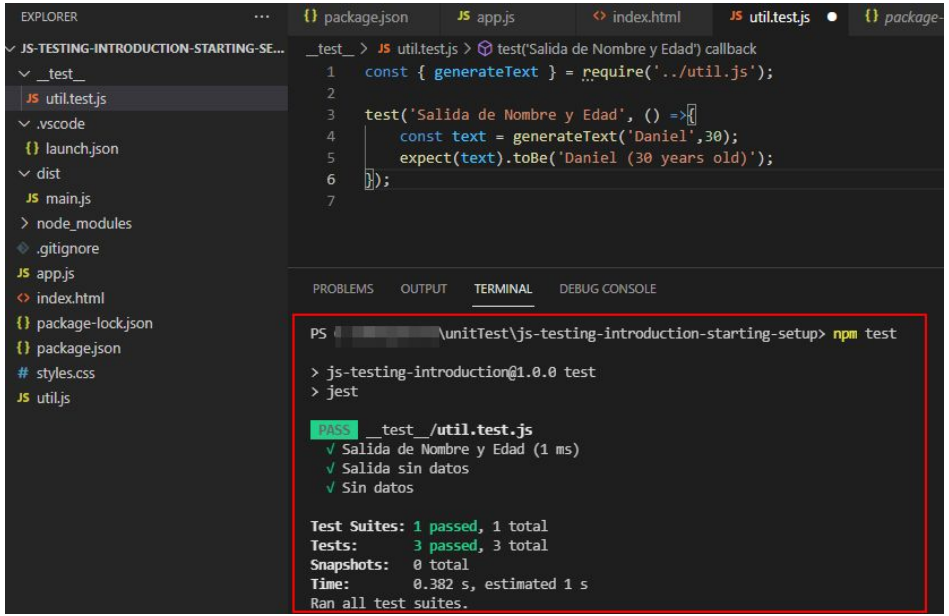
5

Escribir el código del unit test. Las funciones claves a utilizar son:

- **test()** o **it()**: donde se define el test. Se debe ingresar un nombre representativo.
- **expect()**: lo que se desea verificar es parte de la *assertion library*. En el ejemplo se quiere verificar la salida que se genera al presionar el botón **Agregar Usuario**.

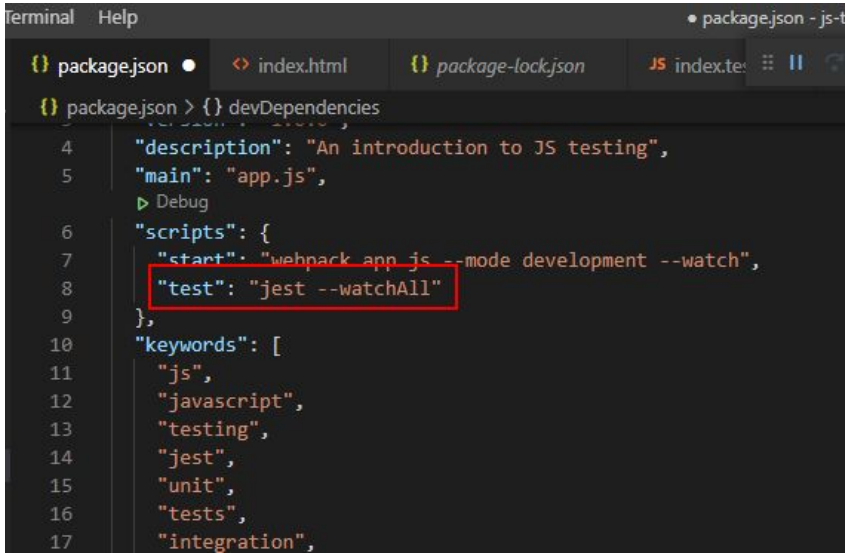
6

Para ejecutar el test, primero debemos abrir la terminal y, luego, ejecutar el comando **npm test**.



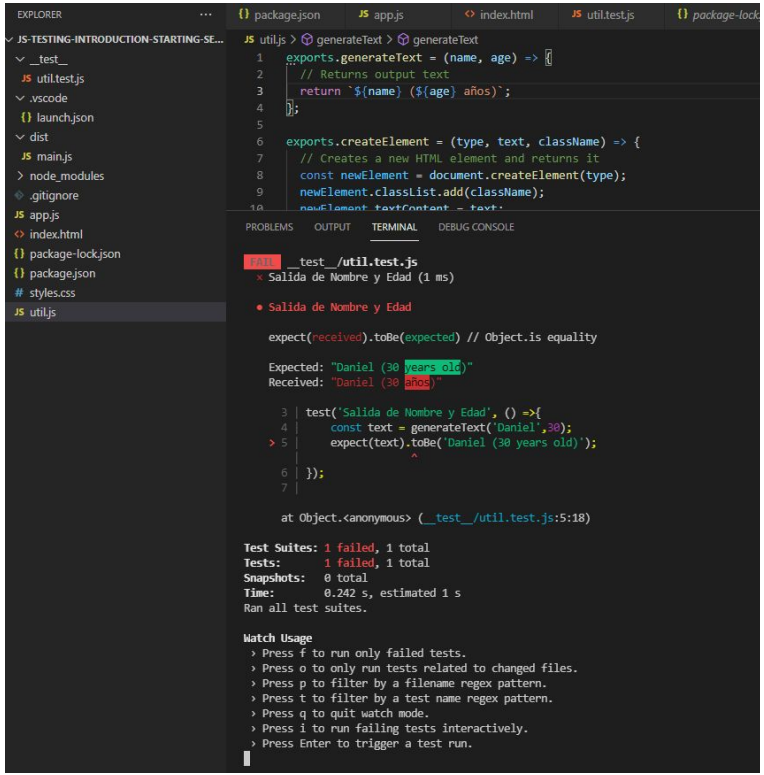
7

Se puede configurar que los tests se ejecuten cada vez que se genera un cambio en el código. Para ello, debemos ir al archivo **package.json** e ingresar **jest --watchAll** en el nodo “test” que está dentro de “scripts”.



Luego, ejecutar el siguiente comando: **npm test**.

Allí, se debe realizar alguna modificación en el código del programa (**util.js**) y, por último, al momento de guardar el archivo modificado, se ejecutarán los tests.



3 Agrupar unit tests

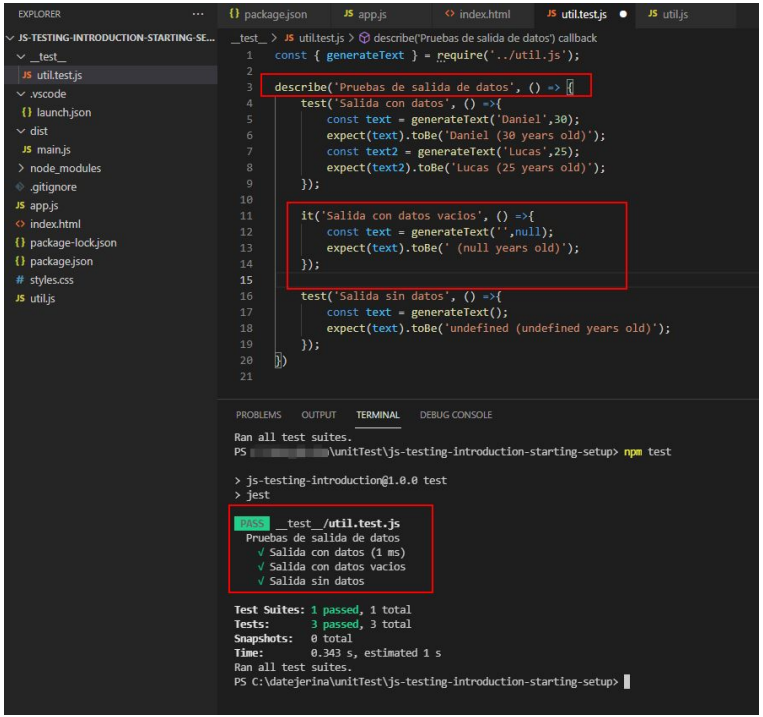
Agrupar unit tests

8

Para generar agrupaciones de unit tests se utiliza la siguiente palabra clave:

describe()

Esto será útil para organizar las pruebas de acuerdo a las funcionalidades a probar.



DigitalHouse>