



GUÍA 2.3.2

Configuración de pruebas unitarias en proyectos front-end

Sigla	Asignatura	Experiencia de Aprendizaje
DSY1104	Desarrollo Full Stack II	EA Configuración de Jasmine y Karma
Tiempo	Modalidad de Trabajo	Indicadores de logro
2 h	Individual	IL 2.2



Antecedentes generales

Esta guía tiene como objetivo enumerar las acciones necesarias para dar solución a los problemas planteados.



Requerimientos para esta actividad

Para el desarrollo de esta actividad deberás disponer de:

- Computador
- AWS – EC2



Actividad

Esta actividad consiste en enumerar las acciones necesarias para dar solución a los casos que se verán a continuación, para ello los estudiantes deberán realizar la actividad de forma individual.

Sigue las Instrucciones

1. Ingresar a EC2 > Instancias y presionar Conectar

The screenshot shows the AWS Management Console interface for the EC2 service. The top navigation bar includes 'Instancias (1/2)', 'Información', and several action buttons: 'C', 'Conectar', 'Estado de la instancia', 'Acciones', and a yellow 'Lan' button. Below the navigation is a search bar with placeholder text 'Buscar Instancia por atributo o etiqueta (case-sensitive)'. The main table lists one instance: 'reactBootstrap' (ID: i-00f383f1bf61d2feb), which is currently 'En ejecución'. To the right of the instance details is another 'Actions' dropdown with options: 'Lanzar instancias', 'Lanzar la instancia desde una plantilla', 'Migrar un servidor', and 'Conectar'. At the bottom of the table, there are additional columns for 'Name', 'ID de la instancia', 'Estado de la i...', 'Tipo de inst...', and a 'Conectar' button.

2. Se abrirá una nueva pestaña de conexión y copia el código que esta enmarcado en rojo:



Conexión de la instancia EC2 Administrador de sesiones **Cliente SSH** Consola de serie de EC2

ID de la instancia
 i-00f383f1bf61d2feb (reactBootstrap)

1. Abra un cliente SSH.
2. Localice el archivo de clave privada. La clave utilizada para lanzar esta instancia es userReact.pem
3. Ejecute este comando, si es necesario, para garantizar que la clave no se pueda ver públicamente.
 chmod 400 "userReact.pem"
4. Conéctese a la instancia mediante su DNS público:
 ec2-3-94-255-181.compute-1.amazonaws.com

Ejemplo:
 ssh -i "userReact.pem" ubuntu@ec2-3-94-255-181.compute-1.amazonaws.com

3. Abre un CMD y ejecútalo como administrador. Debes dirigirte a la dirección donde esta userReact.pem y copiar el ejemplo anterior y colocar yes

```
C:\Users\Donacion\Downloads>ssh -i "userReact.pem" ubuntu@ec2-3-94-255-181.compute-1.amazonaws.com
The authenticity of host 'ec2-3-94-255-181.compute-1.amazonaws.com (3.94.255.181)' can't be established.
ED25519 key fingerprint is SHA256:bmVTooI0d1CwTMq7n4/JSzHQ0DJK165Pa+k9UD0ezaw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes■
```

4. Instalación de dependencias

```
```bash
npm install --save-dev karma karma-jasmine jasmine-core karma-chrome-launcher
````
```

5. Configuración de Karma

```
```javascript
// karma.conf.js
module.exports = function(config) {
 config.set({
 frameworks: ['jasmine'],
 files: [
 'src/**/*.js',
 'test/**/*.js'
],
 browsers: ['ChromeHeadless'],
 singleRun: true
 });
};
```

6. Estructura de una prueba Jasmine

```
```javascript
describe('Componente', () => {
  it('debería hacer algo específico', () => {
    expect(true).toBe(true);
  });
});
```



7. Probando un componente React simple

```
```javascript
import React from 'react';
import { render } from '@testing-library/react';
import MiComponente from './MiComponente';

describe('MiComponente', () => {
 it('renderiza correctamente', () => {
 const { getByText } = render(<MiComponente />);
 expect(getByText('Hola Mundo')).toBeInTheDocument();
 });
});
```
```

```

## 8. Ejemplo de mock con Jasmine

```
```javascript
describe('ServicioAPI', () => {
  let servicioAPI;

  beforeEach(() => {
    servicioAPI = jasmine.createSpyObj('ServicioAPI', ['getData']);
    servicioAPI.getData.and.returnValue(Promise.resolve({ data: 'test' }));
  });

  it('debería llamar a getData', async () => {
    await servicioAPI.getData();
    expect(servicioAPI.getData).toHaveBeenCalled();
  });
});
```
```

```

9. Probando cambios de estado

```
```javascript
import { render, fireEvent } from '@testing-library/react';

test('incrementa el contador cuando se hace clic', () => {
 const { getByText } = render(<Contador />);
 const botón = getByText('Incrementar');
 fireEvent.click(botón);
 expect(getByText('Contador: 1')).toBeInTheDocument();
});
```
```

```

## 10. Simulación de eventos del usuario

```
```javascript
test('llama a onSubmit cuando se envía el formulario', () => {
  const onSubmit = jasmine.createSpy('onSubmit');
  const { getByText } = render(<Formulario onSubmit={onSubmit} />);
```
```

```



```
fireEvent.click(getByText('Enviar'));
expect(onSubmit).toHaveBeenCalled();
});  
```
```

#### 11. Testeando hooks con renderHook

```
```javascript
import { renderHook, act } from '@testing-library/react-hooks';
import useContador from './useContador';

test('debería incrementar el contador', () => {
  const { result } = renderHook(() => useContador());
  act(() => {
    result.current.incrementar();
  });
  expect(result.current.contador).toBe(1);
});  
```
```

#### 12. Mocking de llamadas a API

```
```javascript
import axios from 'axios';
jest.mock('axios');

test('fetches datos exitosamente', async () => {
  const datos = { id: 1, nombre: 'Test' };
  axios.get.mockResolvedValue({ data: datos });
  const result = await fetchDatos();
  expect(result).toEqual(datos);
});  
```
```

#### 13. Configuración de cobertura en Karma

```
```javascript
// karma.conf.js
module.exports = function(config) {
  config.set({
    // ...
    reporters: ['progress', 'coverage'],
    preprocessors: {
      'src/**/*.{js,ts}': ['coverage']
    },
    coverageReporter: {
      type: 'html',
      dir: 'coverage/'
    }
  });
};  
```
```

#### 14. Paralelización de pruebas

```
```javascript
// karma.conf.js
module.exports = function(config) {
  config.set({
    // ...
    concurrency: 2 // Ajustar según recursos disponibles
  });
};  
```
```



```
});
};
};
```

#### 15. Ejemplo de buildspec.yml para CodeBuild

```
```yaml  
version: 0.2  
phases:  
  install:  
    runtime-versions:  
      nodejs: 14  
    commands:  
      - npm install  
  build:  
    commands:  
      - npm test  
...  
```
```

#### 16. Medición de tiempos de renderizado

```
```javascript  
import { render } from '@testing-library/react';  
import { performance } from 'perf_hooks';  
  
test('renderiza rápidamente', () => {  
  const start = performance.now();  
  render(<ComponentePesado />);  
  const end = performance.now();  
  expect(end - start).toBeLessThan(100); // menos de 100ms  
});  
...  
```
```

#### 17. Pruebas de manejo de errores

```
```javascript  
test('maneja errores de API correctamente', async () => {  
  axios.get.mockRejectedValue(new Error('API Error'));  
  await expect(fetchDatos()).rejects.toThrow('API Error');  
});  
...  
```
```

#### 18. Integración de pruebas de accesibilidad

```
```javascript  
import { axe } from 'jest-axe';  
  
test('no tiene violaciones de accesibilidad', async () => {  
  const { container } = render(<MiComponente />);  
  const results = await axe(container);  
  expect(results).toHaveNoViolations();  
});  
...  
```
```

#### 19. Ejemplo de documentación de prueba

```
```javascript  
/**  
 * Prueba la funcionalidad de login  
 * @param {string} username - Nombre de usuario  
 * @param {string} password - Contraseña  
 */  
...  
```
```



```
* @returns {boolean} - true si el login es exitoso
*/
test('realiza login correctamente', () => {
 // ...
});
```

## 20. Configuración de seguridad en EC2

- Asegúrate de que el puerto 3000 esté abierto en el grupo de seguridad de tu instancia EC2.

*```bash*

```
curl http://tu-ip-publica:3000
```

*```*

## 21. Ejecutar la aplicación

- Inicia la aplicación:

*```bash*

```
npm start
```

*```*

- Accede a tu aplicación desde un navegador usando la IP pública de tu instancia EC2: `http://tu-ip-publica:3000`

Nota: Este método es para desarrollo. Para producción, se recomienda construir la aplicación y servirla con un servidor web como Nginx.

## 22. Debería levantar la App

### Recursos de apoyo

- Documentación Create React App: <https://cra.link/deployment>