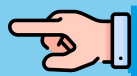
Pruebas Unitarias y end-to-end en un entorno Vue

Herramientas para el testing end-to-end

{desafío} latam_

Implementar pruebas unitarias utilizando las herramientas provistas por Vue para verificar el correcto funcionamiento del aplicativo.

- Unidad 1:
  Vue Router
- Unidad 2:
  Vuex
- Unidad 3:
  Firebase
- Unidad 4:
  Pruebas Unitarias y end-to-end en un entorno Vue



Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Reconoce los conceptos y herramientas utilizados para la realización de pruebas end-to-end en un entorno Vue.*

¿Qué podemos simular al realizar test con Cypress?

{desafío} latam_

Contexto antes de iniciar

Para la ejecución de pruebas end to end utilizamos Cypress, recordemos que este tipo de pruebas nos sirven para simular la interacción del usuario, verificar la integración de los componentes que conforman una aplicación web, etc.

En esta sesión seguiremos trabajando con este tipo de pruebas y conoceremos el framework Nightwatch y sus diferencias con Cypress.

{desafío} latam_

/* Nightwatch */

¿Qué es Nightwatch?

Es un marco de automatización de pruebas de extremo a extremo (E2E), se utiliza para realizar pruebas en aplicaciones web. Con esta tecnología podemos probar desde la interfaz de usuario hasta las funcionalidades del servidor.

Ventajas de Nightwatch

- **Automatización:** Permite automatizar pruebas para el ahorro de tiempo.
- **Facilidad de implementación:** Posee una sintaxis sencilla y está basado en JavaScript.
- **Multi browsers:** Tiene soporte en múltiples navegadores web.
- **Informes detallados:** Resultados detallados de pruebas exitosas y fallidas.

Desventajas de Nightwatch

- **Curva de aprendizaje:** Se requieren conocimientos previos básicos en JavaScript.
- **Hardware:** Correr pruebas con esta tecnología puede requerir una cantidad significativa de recursos de hardware del computador.
- **Compatibilidad constante:** Aunque Nightwatch tenga compatibilidad con diversos navegadores, dado que estos cambian de manera rápida, puede que Nightwatch tenga algunos problemas, esta sería una desventaja de dependencias externas más que de la propia tecnología.

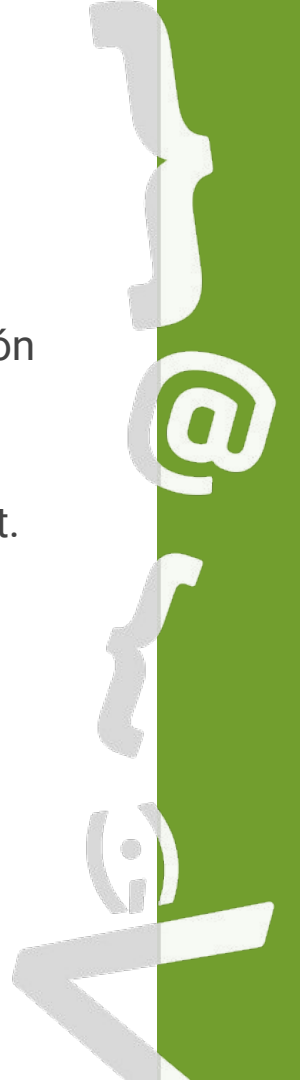
Demostración: "Integrando Nightwatch en Vue JS"

{desafío} latam_

Integrando Nightwatch en Vue JS

A continuación, realizaremos un ejercicio donde veremos el proceso de integración de Nightwatch en una aplicación web con Vue JS. Posteriormente, veremos la estructura de directorios que se genere para identificar una de las cualidades de Nightwatch al definirse como un marco de trabajo para la implementación de test.

Sigue los pasos…



Sigues los pasos…

- **Paso 1:** Creamos la aplicación con el CLI de Vue con el nombre integrando-nightwatch. `npm create vue@latest`
- **Paso 2:** Seguimos esta configuración y seleccionamos Nightwatch en la opción de End-to-End Testing.

```
Vue.js - The Progressive JavaScript Framework

✔ Project name: … integrando-nightwatch
✔ Add TypeScript? … No / Yes
✔ Add JSX Support? … No / Yes
✔ Add Vue Router for Single Page Application development? … No / Yes
✔ Add Pinia for state management? … No / Yes
✔ Add Vitest for Unit Testing? … No / Yes
? Add an End-to-End Testing Solution? › - Use arrow-keys. Return to submit.
    No
    Cypress
❯   Nightwatch - also supports unit testing with Nightwatch Component Testing
    Playwright
```

Sigues los pasos…

- **Paso 3:** La configuración final debe quedar así.

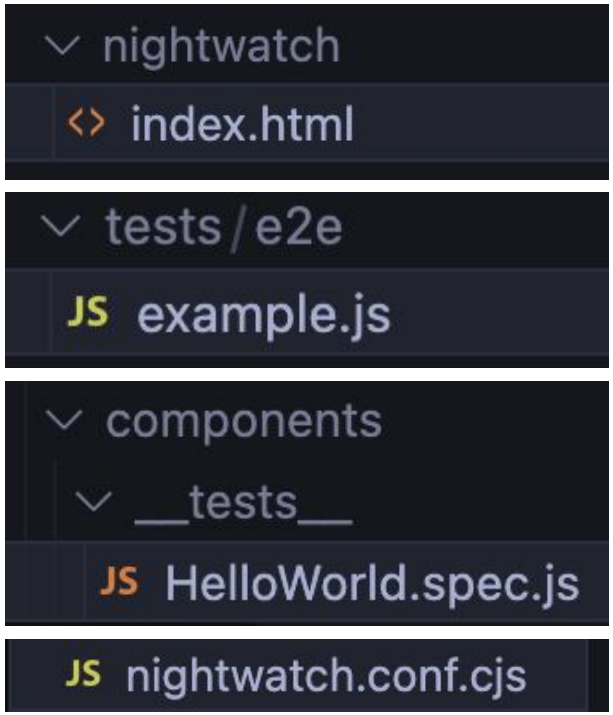
```
Vue.js - The Progressive JavaScript Framework

✔ Project name: … integrando-nightwatch
✔ Add TypeScript? … No / Yes
✔ Add JSX Support? … No / Yes
✔ Add Vue Router for Single Page Application development? … No / Yes
✔ Add Pinia for state management? … No / Yes
✔ Add Vitest for Unit Testing? … No / Yes
✔ Add an End-to-End Testing Solution? › Nightwatch
✔ Add ESLint for code quality? … No / Yes
✔ Add Prettier for code formatting? … No / Yes
✔ Add Vue DevTools 7 extension for debugging? (experimental) … No / Yes
```

Validación de dependencias

- **Paso 4:** Instalar dependencias con `npm install`

```
{
  "name": "integrando-nightwatch",
  "version": "0.0.0",
  "private": true,
  "type": "module",
  ▷ Debug
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview",
    "test:e2e": "nightwatch tests/e2e/*",
    "test:unit": "nightwatch src/**/__tests__/*",
    "lint": "eslint . --ext .vue,.js,.jsx,.cjs,.mjs --fix --ignore-path .gitignore",
    "format": "prettier --write src/"
  },
  "dependencies": {
    "vue": "^3.4.29"
  },
  "devDependencies": {
    "@nightwatch/vue": "^3.1.1",
    "@rushstack/eslint-patch": "^1.8.0",
    "@vitejs/plugin-vue": "^5.0.5",
    "@vue/eslint-config-prettier": "^9.0.0",
    "@vue/test-utils": "^2.4.6",
    "chromedriver": "^126.0.2",
    "eslint": "^8.57.0",
    "eslint-plugin-vue": "^9.23.0",
    "geckodriver": "^4.4.1",
    "nightwatch": "^3.6.3",
    "prettier": "^3.2.5",
    "ts-node": "^10.9.2",
    "vite": "^5.3.1",
    "vite-plugin-nightwatch": "^0.4.6"
  }
}
```

Carpeta de tests



Archivo de configuración

En el directorio raiz de la aplicación creamos un archivo con el nombre nightwatch.conf.js y agregamos el siguiente código.

```
// nightwatch.conf.js
module.exports = {
    src_folders: ["tests"],
    webdriver: {
      start_process: true,
      server_path: require("chromedriver").path,
      port: 9515,
    },
    test_settings: {
      default: {
        desiredCapabilities: {
          browserName: "chrome",
        },
      },
    },
  };
```

Resultado de ejecución de prueba

En el directorio `/tests/e2e/` el archivo `example.js` contiene las pruebas y aserciones.

tests > e2e > JS example.js > ...

```
describe('My First Test', function () {
  before((browser) => {
    browser.init()
  })

  it('visits the app root url', function () {
    browser.assert.textContains('.green', 'You did it!')
  })

  after((browser) => browser.end())
})
```

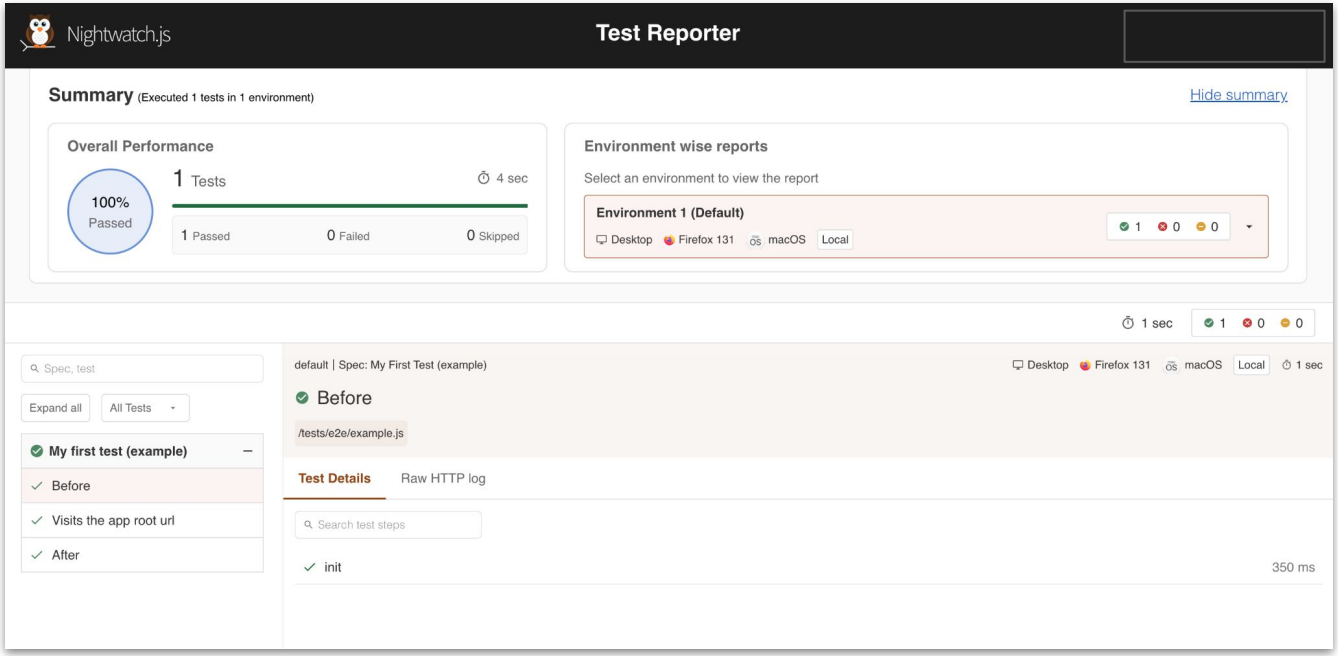
Ejecútala con `npm run test:e2e`

Resultado de ejecución de prueba

```
> integrando-nightwatch@0.0.0 test:e2e
> nightwatch tests/e2e/*


[My First Test] Test Suite
──────────────────────────

ℹ Connected to GeckoDriver on port 4444 (2140ms).
  Using: firefox (131.0) on MAC.

  ℹ Loaded url http://localhost:5173 in 265ms

  Running visits the app root url:
  ──────────────────────────────────

  ✔ Testing if element <.green> contains text 'You did it!' (63ms)

  ✨ PASSED. 1 assertions. (71ms)
 Wrote HTML report file to: /                                    e/integrando-nightwatch/tests_output/nightwatch-html-report/index.html
```

Resultado de ejecución de prueba

Puedes notar al final que indica que ha generado un reporte en un archivo HTML, si lo abres veremos algo así



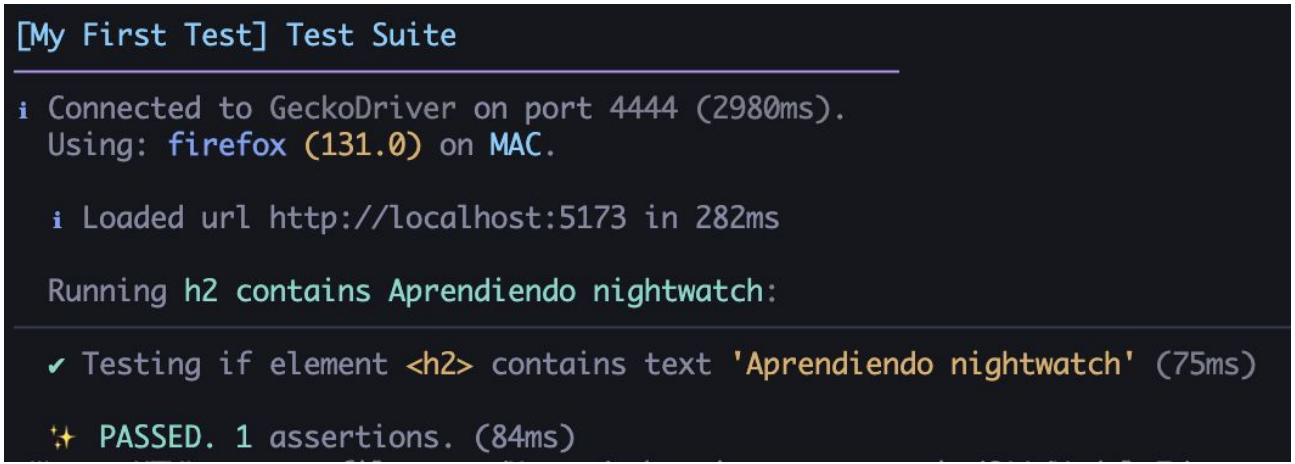
{desafío} latam_

Ejercicio:
"Realiza tu primera prueba en Nightwatch"

Realiza tu primera prueba en Nightwatch

Contexto

A continuación, utilizando de ejemplo la aplicación del ejercicio anterior, crea tu primera prueba en Nightatch. Para lograrlo crea un h2 en el componente principal App.vue con el texto “Aprendiendo nightwatch”, luego escribe la prueba que evalúe que el h2 contenga el texto solicitado, al correr esta prueba verás el siguiente resultado



{desafío} latam_

Resumen de la sesión

- En esta sesión conocimos Nightwatch, un marco de ejecución de pruebas e2e.
- Este marco de pruebas nos provee diversas funcionalidades como evaluar la existencia de elementos o texto.
- La interacción de Nightwatch es directamente con el navegador y tiene múltiples soportes.

¿Existe algún concepto que requieras repasar?

Próxima sesión...

- *Guía de ejercicios.*

{desafío}
latam_

{desafío}
latam_

Academia de
talentos digitales