

# Testing y Calidad

## Presentado por:

Juan José Martín Vargas – [jmartinv@unal.edu.co](mailto:jmartinv@unal.edu.co)

Juan Camilo Posso Portilla – [jpossop@unal.edu.co](mailto:jpossop@unal.edu.co)

Esteban Prieto Lugo - [eprietol@unal.edu.co](mailto:eprietol@unal.edu.co)

Juan Camilo Vergara Tao – [juvergarat@unal.edu.co](mailto:juvergarat@unal.edu.co)

## Profesor:

Oscar Eduardo Alvarez Rodriguez

[oalvarezr@unal.edu.co](mailto:oalvarezr@unal.edu.co)



**Universidad Nacional de Colombia**

**Facultad de Ingeniería**

**Departamento de Ingeniería de Sistemas y Computación**

**2025-1S**

# Testing

Este documento detalla la implementación y los resultados de las pruebas unitarias para el proyecto. El objetivo de estas pruebas fue validar la **lógica de negocio esencial** del sistema, asegurando la fiabilidad y correctitud de las funciones de cálculo y validación de datos.

Para la ejecución de las pruebas se utilizó el framework **Vitest**, una herramienta de testing moderna y rápida para proyectos de Vite. Las pruebas se ejecutan con el comando **npm run test -- --reporter verbose**.

## Pruebas Unitarias

La estrategia de pruebas se centró en el archivo `src/lib/__tests__/utils.test.js`, que contiene **12 pruebas** diseñadas y documentadas para cubrir exhaustivamente las funciones críticas de la aplicación. A continuación se presenta una explicación de cada prueba:

### Pruebas para calculoIMC

- **calcula IMC correctamente para valores normales:** Esta prueba verifica que la función `calculoIMC` calcula el Índice de Masa Corporal (IMC) de forma precisa cuando se le proporcionan valores de peso y altura que están dentro de un rango típico y esperado.
- **maneja strings como entrada:** Esta prueba asegura que la función `calculoIMC` es lo suficientemente robusta como para procesar y calcular el IMC correctamente incluso si el peso y la altura se le pasan como cadenas de texto en lugar de números.

### Pruebas para calcularCalorias

- **calcula calorías correctamente para hombre que mantiene peso:** Este test comprueba que la función `calcularCalorias` determina de manera correcta las calorías diarias necesarias para un hombre que busca mantener su peso actual, basándose en los datos de entrada proporcionados.
- **calcula calorías correctamente para mujer que mantiene peso:** De forma similar, esta prueba verifica que `calcularCalorias` calcula adecuadamente las calorías diarias para una mujer cuyo objetivo es mantener su peso, utilizando los datos de entrada específicos para ella.

### Pruebas para validarCampos

- **valida correctamente campos válidos:** Esta prueba confirma que la función `validarCampos` retorna que todos los campos son válidos cuando se le proporciona un conjunto de datos que cumplen con todas las condiciones de validación.
- **rechaza edad menor a 18:** Este test verifica que `validarCampos` identifica y marca como inválida una entrada donde la edad es inferior a 18 años, devolviendo el mensaje de error apropiado.

- **rechaza edad mayor a 100:** Esta prueba asegura que validarCampos también rechaza una entrada si la edad es superior a 100 años, indicando el mensaje de error correcto para este caso.
- **devuelve el primer error encontrado:** Esta prueba valida que la función validarCampos es eficiente al detenerse y reportar solo el primer error que encuentra en la validación de múltiples campos, sin procesar los errores subsiguientes.

## Pruebas para validarPassword

- **acepta contraseñas válidas:** Esta prueba verifica que la función validarPassword reconoce y aprueba contraseñas que cumplen con todos los criterios de seguridad establecidos (por ejemplo, longitud mínima, presencia de mayúsculas, etc.).
- **rechaza contraseñas muy cortas:** Este test confirma que validarPassword niega el acceso a contraseñas que no alcanzan la longitud mínima requerida, asegurando que las contraseñas cortas sean rechazadas por seguridad.
- **rechaza contraseñas sin mayúsculas:** Esta prueba asegura que validarPassword considera inválidas las contraseñas que no contienen al menos una letra mayúscula, reforzando así la complejidad de las contraseñas aceptadas.
- **rechaza contraseña vacía o nula:** Este test es crucial para verificar que la función validarPassword maneja correctamente casos extremos, rechazando explícitamente cualquier contraseña que sea nula, indefinida o una cadena de texto vacía.

## Resultados

La ejecución de la suite de pruebas arrojó un resultado exitoso, confirmando que toda la funcionalidad cubierta opera según lo esperado. El reporte detallado (verbose) muestra el desglose de cada prueba superada, de la siguiente manera.

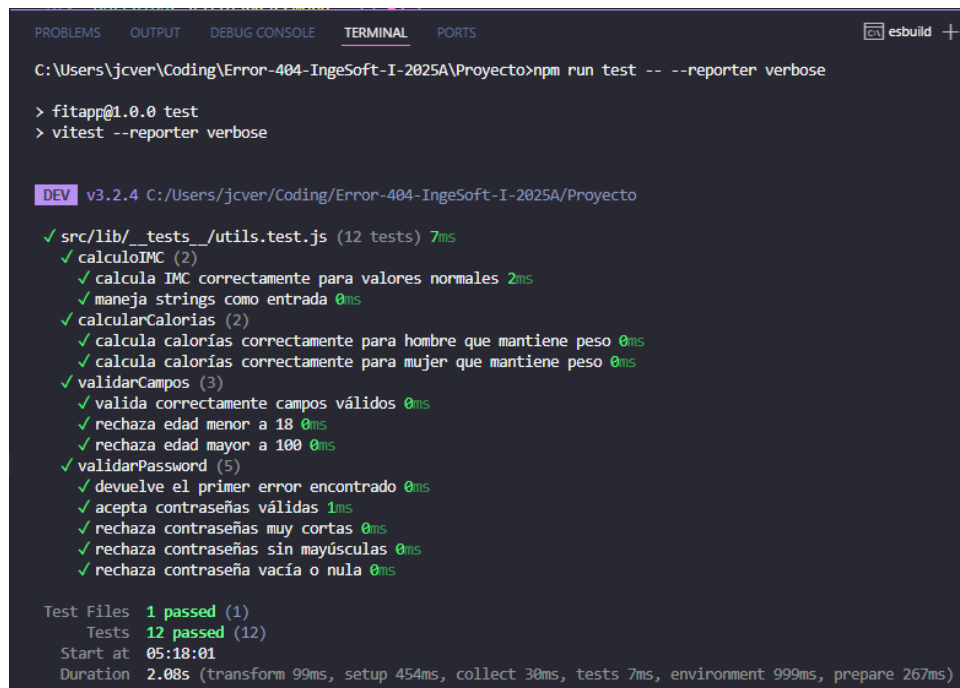
```

C:\Users\Jover\Idea\Error-404-IngenSoft-1-2025A\Proyecto> npm test -- --reporter verbose
> fitaggit-0.8 test
> vitest --reporter verbose

v3.2.4 C:\Users\Jover\Idea\Error-404-IngenSoft-1-2025A\Proyecto
src/lib/_tests_/utils.test.js (12 tests) 3ms
  ✓ calculoIMC (2)
    ✓ calcula IMC correctamente para valores normales 2ms
    ✓ devuelve string como entrada 0ms
  ✓ calcularCalorias (2)
    ✓ calcula calorias correctamente para hombre que mantiene peso 0ms
    ✓ calcula calorias correctamente para mujer que mantiene peso 0ms
  ✓ validarCampos (2)
    ✓ valida correctamente campos validos 0ms
    ✓ rechaza edad menor a 18 0ms
  ✓ rechaza edad mayor a 100 0ms
  ✓ validarPassword (2)
    ✓ devuelve el primer error encontrado 0ms
    ✓ acepta contraseñas validas 0ms
  ✓ rechaza contraseñas muy cortas 0ms
  ✓ rechaza contraseñas sin mayúsculas 0ms
  ✓ rechaza contraseña vacía o nula 0ms

Test Files 1 passed (1)
Tests 12 passed (12)
Start at 05:18:01
Duration 2.88s (transform 99ms, setup 454ms, collect 38ms, tests 7ms, environment 599ms, prepare 262ms)

Waiting for file changes...
press h to show help, press q to quit
  
```



```
C:\Users\jcver\Coding\Error-404-IngeSoft-I-2025A\Proyecto>npm run test -- --reporter verbose


> fitapp@1.0.0 test
> vitest --reporter verbose

DEV v3.2.4 C:\Users\jcver\Coding\Error-404-IngeSoft-I-2025A\Proyecto

✓ src/lib/_tests_/utils.test.js (12 tests) 7ms
  ✓ calculoIMC (2)
    ✓ calcula IMC correctamente para valores normales 2ms
    ✓ maneja strings como entrada 0ms
  ✓ calcularCalorias (2)
    ✓ calcula calorías correctamente para hombre que mantiene peso 0ms
    ✓ calcula calorías correctamente para mujer que mantiene peso 0ms
  ✓ validarCampos (3)
    ✓ valida correctamente campos válidos 0ms
    ✓ rechaza edad menor a 18 0ms
    ✓ rechaza edad mayor a 100 0ms
  ✓ validarPassword (5)
    ✓ devuelve el primer error encontrado 0ms
    ✓ acepta contraseñas válidas 1ms
    ✓ rechaza contraseñas muy cortas 0ms
    ✓ rechaza contraseñas sin mayúsculas 0ms
    ✓ rechaza contraseña vacía o nula 0ms

Test Files  1 passed (1)
Tests       12 passed (12)
Start at    05:18:01
Duration    2.08s (transform 99ms, setup 454ms, collect 30ms, tests 7ms, environment 999ms, prepare 267ms)
```

### Resumen de Resultados:

- **Archivos de Prueba:** 1 de 1 pasó.
- **Pruebas Totales:** 12 de 12 pasaron.
- **Estado:**  **Éxito.**

La implementación de esta suite de pruebas unitarias con Vitest ha sido un paso fundamental para garantizar la calidad y estabilidad del proyecto. Al validar con éxito la lógica de negocio central, se establece una base de código fiable y robusta, lo que facilita el mantenimiento y la implementación de nuevas funcionalidades a futuro.

## Calidad

Para garantizar la calidad, legibilidad y mantenibilidad del código fuente del proyecto, se implementó un proceso de análisis estático. Este enfoque permite identificar problemas potenciales, errores de estilo y violaciones de buenas prácticas de manera automatizada antes de que el software llegue a una fase de ejecución o pruebas funcionales.

## Herramienta Utilizada

La herramienta principal seleccionada para el análisis estático del código fue **ESLint**, un linter de código abierto ampliamente adoptado en la comunidad de desarrollo de JavaScript.

Dado que el proyecto fue desarrollado con el framework Svelte, se utilizó el plugin **eslint-plugin-svelte3**. Este complemento es esencial, ya que extiende las capacidades de ESLint para que pueda analizar correctamente la sintaxis específica de los archivos `.svelte`, incluyendo tanto el bloque `<script>` (JavaScript) como la estructura HTML y los estilos `<style>`.

## Configuración Aplicada

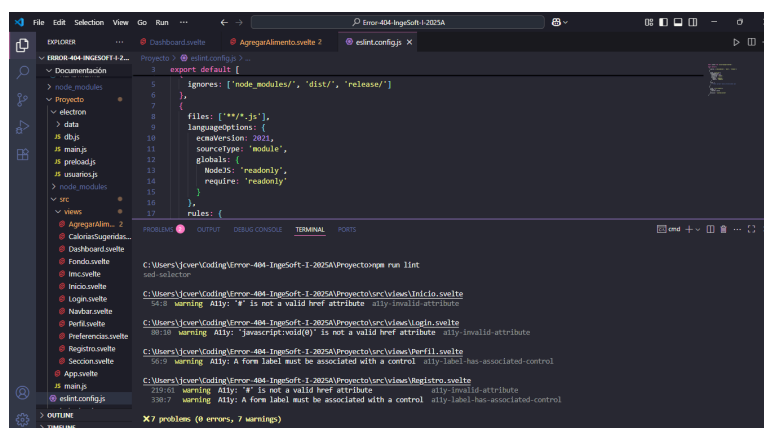
La configuración de ESLint se definió en el archivo **eslint.config.js**. Las directivas más importantes aplicadas al proyecto fueron:

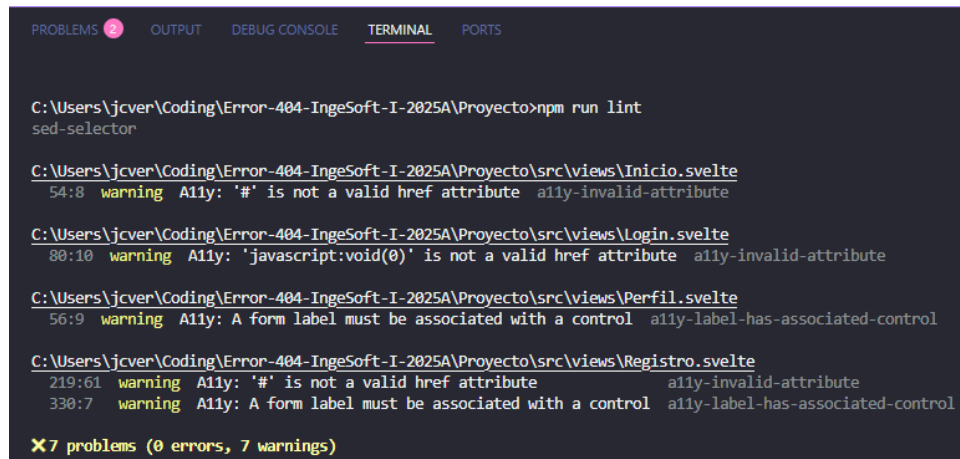
- **Archivos Ignorados:** Se excluyeron del análisis los directorios **node\_modules/** (parte de los frameworks utilizados) y **dist/ y release/** (dada la sugerencia de GitHub Copilot) para evitar analizar código de terceros o archivos generados automáticamente durante la compilación.
- **Análisis de Archivos JavaScript (.js):** Se configuró ESLint para validar los archivos JavaScript bajo el estándar **ECMAScript 2021**.
- **Análisis de Componentes Svelte (.svelte):** Se habilitó el procesador **svelte3/svelte3** del plugin para que ESLint pudiera interpretar y validar la estructura completa de los componentes Svelte.
- **Reglas:** No se agregaron reglas personalizadas adicionales, por lo que el análisis se basó en el conjunto de reglas recomendadas por defecto en ESLint y **eslint-plugin-svelte3**. Esto incluye un conjunto robusto de reglas orientadas a prevenir errores comunes y a fomentar buenas prácticas, especialmente en materia de accesibilidad (A11y).

La verificación del código se ejecutó a través del comando **npm run lint**, que invoca el script **eslint . --ext .js,.svelte** para analizar todos los archivos relevantes en el proyecto.

## Resultados

Tras la ejecución del análisis estático el día 16 de julio de 2025 a las 9:15 pm, **no se reportaron errores críticos** que impidieron la funcionalidad del aplicativo. Sin embargo, se identificaron un total de **7 advertencias (warnings)**, las cuales señalan oportunidades de mejora principalmente en accesibilidad web (A11y) y optimización de código.





```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

C:\Users\jcver\Coding\Error-404-IngeSoft-I-2025A\Proyecto>npm run lint
sed-selector

C:\Users\jcver\Coding\Error-404-IngeSoft-I-2025A\Proyecto\src\views\Inicio.svelte
54:8 warning A11y: '#' is not a valid href attribute a11y-invalid-attribute

C:\Users\jcver\Coding\Error-404-IngeSoft-I-2025A\Proyecto\src\views\Login.svelte
80:10 warning A11y: 'javascript:void(0)' is not a valid href attribute a11y-invalid-attribute

C:\Users\jcver\Coding\Error-404-IngeSoft-I-2025A\Proyecto\src\views\Perfil.svelte
56:9 warning A11y: A form label must be associated with a control a11y-label-has-associated-control

C:\Users\jcver\Coding\Error-404-IngeSoft-I-2025A\Proyecto\src\views\Registro.svelte
219:61 warning A11y: '#' is not a valid href attribute a11y-invalid-attribute
330:7 warning A11y: A form label must be associated with a control a11y-label-has-associated-control

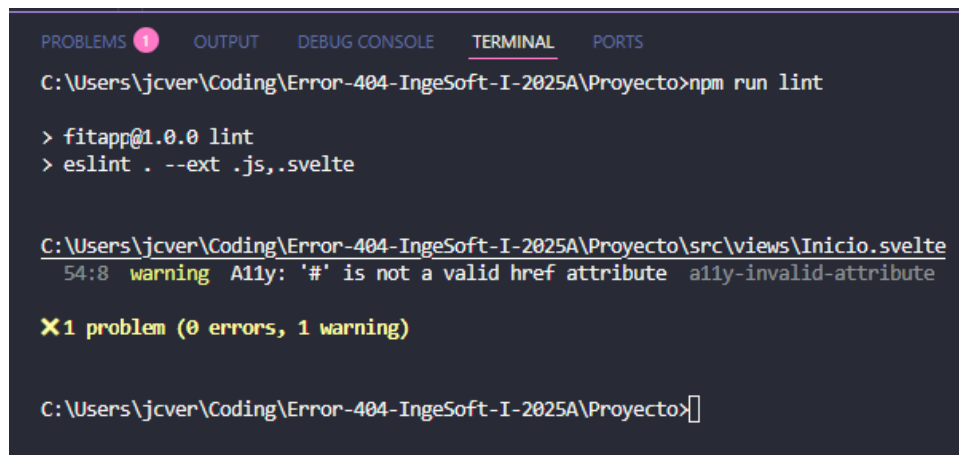
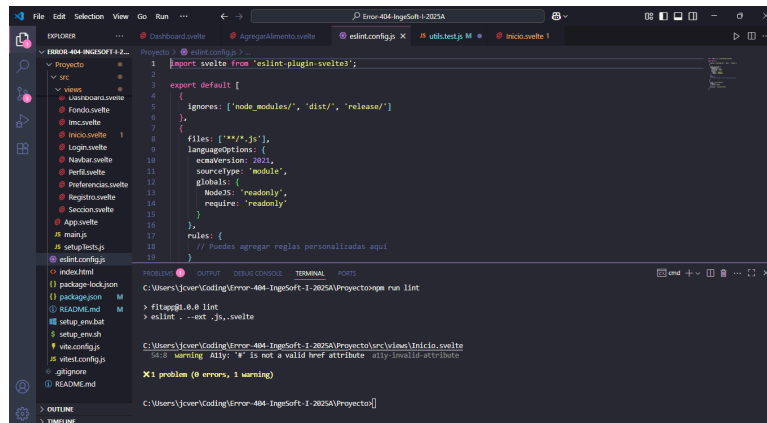
X 7 problems (0 errors, 7 warnings)
```

A continuación, se resumen las advertencias más relevantes encontradas:

- **a11y-invalid-attribute:** Se detectó el uso de valores no válidos en el atributo href de etiquetas <a>, como href="#" o href="javascript:void(0)". Esta práctica es desaconsejada porque puede causar comportamientos inesperados en la navegación y afectar a los lectores de pantalla.
  - Archivos afectados: Inicio.svelte, Login.svelte, Registro.svelte.
- **a11y-label-has-associated-control:** Se encontraron etiquetas <label> que no estaban correctamente asociadas a un campo de formulario (<input>, <select>, etc.). Una correcta asociación es fundamental para que los usuarios de tecnologías de asistencia puedan interactuar eficientemente con los formularios.
  - Archivos afectados: Perfil.svelte, Registro.svelte.
- **a11y-click-events-have-key-events:** Se identificó un elemento visible no interactivo (como un <div> o <span>) con un evento on:click. Para ser accesible, este tipo de elementos debe tener también un evento de teclado equivalente (como on:keydown), permitiendo su activación a usuarios que no utilizan un ratón.
  - Archivo afectado: AgregarAlimento.svelte.
- **css-unused-selector:** Se encontró una regla de estilo CSS (.modal-header h5) que no se está aplicando a ningún elemento en el componente, lo que la convierte en código muerto que puede ser eliminado para optimizar el archivo.
  - Archivo afectado: AgregarAlimento.svelte.

Estas advertencias, aunque no bloquean la ejecución, son importantes para mejorar la experiencia de usuario, especialmente para personas con discapacidades, y para mantener una base de código limpia y optimizada.

Posteriormente se realizó una revisión del código y se ejecutó otro análisis estático el día 19 de julio a las 4:35 am, en el cual los resultados demuestran una mejora sustancial en la calidad del código. El proceso de corrección fue exitoso, resolviendo la mayoría de las advertencias reportadas previamente y dejando el proyecto con un solo señalamiento de bajo impacto.



En la última revisión, se reportaron **cero errores críticos** y **una única advertencia** (warning).

- **Advertencia: a11y-invalid-attribute**
  - **Descripción:** Se identificó el uso del atributo href="#" en una etiqueta de ancla <a>. Esta práctica no es recomendada ya que interfiere con las tecnologías de asistencia y las funcionalidades de navegación del navegador.
  - **Archivo afectado:** Inicio.svelte.

Este resultado demuestra que el proyecto cumple con altos estándares de calidad y buenas prácticas, presentando únicamente un punto menor de mejora relacionado con la accesibilidad web.