

# 

# **Pruebas de mutantes**

1. Miranda Eugenia Colorado Arróniz A01737027
2. Estefania Antonio Villaseca A01736897
3. Alejandro Kong Montoya A01734271
4. Sofia Zugasti Delgado A00837478
5. 12 de mayo de 2025
7. **Desarrollo e implantación de sistemas de software (Gpo 502)**
8. Prof. Juan Manuel González Calleros

Con el propósito de fortalecer la calidad de las pruebas automatizadas mediante la técnica de mutation testing manual, seleccionamos cuatro funciones clave de nuestro proyecto que contienen lógica condicional y cálculos representativos. Estas funciones ya contaban con pruebas automatizadas básicas, por lo que resultaron ideales para aplicar mutaciones y evaluar la robustez del conjunto actual de pruebas.

Los módulos elegidos fueron:

1. **getBackgroundColor(hours: number): string:** Función utilizada para determinar el color visual de una tarjeta en función de la cantidad de horas de un proyecto solidario.
2. **normalize(steps: Step[]): Step[]:** Función encargada de calcular el estado de inscripción de un estudiante según la secuencia de pasos completados o rechazados en un flujo de postulación.
3. **validateForm(formData):** Función que valida los campos de un formulario de postulación, incluyendo nombre, matrícula, correo y compromiso.
4. **matchesSearch(project, search):** Función que evalúa si un proyecto coincide con el término ingresado en la barra de búsqueda, verificando coincidencias en el nombre y objetivo del proyecto.

A continuación se presenta una tabla con los 40 archivos mutados, el tipo de mutación y la descripción del cambio aplicado manualmente a cada función del proyecto:

| **Archivo mutado** | **Tipo de mutación** | **Descripción** |
| --- | --- | --- |
| getBackgroundColor.m1.ts | Valor constante | "bg-teal-300" por "bg-purple-400" |
| getBackgroundColor.m2.ts | Valor constante | "bg-purple-400" por "bg-teal-300" |
| getBackgroundColor.m3.ts | Valor constante | "bg-rose-400" por "bg-orange-400" |
| getBackgroundColor.m4.ts | Valor constante | "bg-orange-400" por "bg-sky-400" |
| getBackgroundColor.m5.ts | Valor constante | "bg-sky-400" por "bg-rose-400" |
| getBackgroundColor.m6.ts | Retorno fijo | return "bg-black" |
| getBackgroundColor.m7.ts | Condición agregada | if (hours < 100) return "bg-yellow-400" |
| getBackgroundColor.m8.ts | Condición modificada | if (hours >= 180) return "bg-lime-400" |
| getBackgroundColor.m9.ts | Condición con negación | if (hours !== 60) return "bg-indigo-400" |
| getBackgroundColor.m10.ts | Validación de tipo | if (typeof hours !== "number") |
| normalize.m1.ts | Estado por defecto | insStatus = "completed" |
| normalize.m2.ts | Estado por defecto | insStatus = "rejected" |
| normalize.m3.ts | Lógica invertida | if (resp === "completed") insStatus = "rejected" |
| normalize.m4.ts | Lógica invertida | if (resp === "rejected") insStatus = "completed" |
| normalize.m5.ts | Omisión de paso | No se agrega paso "Inscritx" |
| normalize.m6.ts | Retorno base | return base |
| normalize.m7.ts | Retorno vacío | return [] |
| normalize.m8.ts | Forzar estado | base[1].status = "rejected" |
| normalize.m9.ts | Cambiar todos los estados | status = "completed" |
| normalize.m10.ts | Agregar paso fijo | Agregar Inscritx sin condición |
| validateForm.m1.ts | Validación eliminada | No se usa .trim() en nombre |
| validateForm.m2.ts | Dominio cambiado | email incluye ".com" |
| validateForm.m3.ts | Longitud modificada | número < 10 |
| validateForm.m4.ts | Longitud modificada | número > 10 |
| validateForm.m5.ts | Email vacío | email === "" |
| validateForm.m6.ts | Longitud matrícula | matrícula.length < 5 |
| validateForm.m7.ts | Sin .trim() | No se usa .trim() en carrera\_completa |
| validateForm.m8.ts | Longitud nombre | nombre.length < 2 |
| validateForm.m9.ts | Dominio estricto | email incluye "@tec.mx" |
| validateForm.m10.ts | Retorno fijo | return {} |
| matchesSearch.m1.ts | Comparación exacta | project.proyecto === search |
| matchesSearch.m2.ts | Sin lowerCase | project.objetivo\_ps.includes(search) |
| matchesSearch.m3.ts | Retorno fijo | return false |
| matchesSearch.m4.ts | Retorno fijo | return true |
| matchesSearch.m5.ts | Cambio de método | startsWith en proyecto |
| matchesSearch.m6.ts | Cambio de método | endsWith en objetivo\_ps |
| matchesSearch.m7.ts | Comparación vacía | project.proyecto === "" |
| matchesSearch.m8.ts | Comparación vacía | project.objetivo\_ps === "" |
| matchesSearch.m9.ts | Negación | return !search |
| matchesSearch.m10.ts | Comparación lowerCase | proyecto.toLowerCase() === searchLower |

Este documento contiene el registro de las 40 mutaciones aplicadas manualmente a 4 funciones del proyecto. Cada mutación fue documentada especificando el tipo de error simulado, y se deja espacio para anotar los resultados de las pruebas automatizadas correspondientes (si fallaron o no) y observaciones sobre qué mejorar en cada caso.

**Mutantes de getBackgroundColor**

| **Mutante** | **Tipo de error simulado** | **¿Falló la prueba?** | **¿Qué mejorar en la prueba?** |
| --- | --- | --- | --- |
| m1 | Mutación de color 180 – Cambiar "bg-teal-300" por "bg-purple-400" | Sí | - |
| m2 | Mutación de color 120 – Cambiar "bg-purple-400" por "bg-teal-300" | Sí | - |
| m3 | Mutación de color 60 – Cambiar "bg-rose-400" por "bg-orange-400" | No | Agregar test específico para este caso |
| m4 | Mutación de color 100 – Cambiar "bg-orange-400" por "bg-sky-400" | Sí | - |
| m5 | Mutación de color 200 – Cambiar "bg-sky-400" por "bg-rose-400" | Sí | - |
| m6 | Retorno final fijo – Cambiar return final por "bg-black" | No | Agregar test específico para este caso |
| m7 | Condición menor que 100 – Agregar condición: if (hours < 100) return "bg-yellow-400" | Sí | - |
| m8 | Operador mayor o igual – Agregar condición: if (hours >= 180) return "bg-lime-400" | Sí | - |
| m9 | Negación de igualdad – Agregar condición: if (hours !== 60) return "bg-indigo-400" | No | Agregar test específico para este caso |
| m10 | Validación de tipo – Agregar if typeof !== number | Sí | - |

**Mutantes de normalize**

| **Mutante** | **Tipo de error simulado** | **¿Falló la prueba?** | **¿Qué mejorar en la prueba?** |
| --- | --- | --- | --- |
| m1 | Status por defecto completado – Cambiar status por defecto a "completed" | Sí | - |
| m2 | Status por defecto rechazado – Cambiar status por defecto a "rejected" | Sí | - |
| m3 | Invertir lógica completed – Cambiar lógica: if completed → rejected | No | Agregar test específico para este caso |
| m4 | Invertir lógica rejected – Cambiar lógica: if rejected → completed | Sí | - |
| m5 | Eliminar paso Inscritx – No se agrega paso Inscritx | Sí | - |
| m6 | Retornar base sin cambios – return base | No | Agregar test específico para este caso |
| m7 | Retornar arreglo vacío – return [] | Sí | - |
| m8 | Forzar cambio de estado – base[1].status = "rejected" | Sí | - |
| m9 | Marcar todos como completed – for all i >= 2 → status = completed | No | Agregar test específico para este caso |
| m10 | Agregar paso fijo Inscritx – push sin condicional | Sí | - |

**Mutantes de validateForm**

| **Mutante** | **Tipo de error simulado** | **¿Falló la prueba?** | **¿Qué mejorar en la prueba?** |
| --- | --- | --- | --- |
| m1 | Eliminar .trim() – Validar nombre sin usar .trim() | Sí | - |
| m2 | Validar .com – Cambiar includes("@") por includes(".com") | Sí | - |
| m3 | Número muy corto – Validar número con < 10 | No | Agregar test específico para este caso |
| m4 | Número muy largo – Agregar validación > 10 | Sí | - |
| m5 | Email vacío – Comparar email === "" | Sí | - |
| m6 | Matrícula corta – Agregar validación de longitud < 5 | No | Agregar test específico para este caso |
| m7 | Sin carrera\_completa – Eliminar trim() de carrera\_completa | Sí | - |
| m8 | Nombre muy corto – Validar nombre.length < 2 | Sí | - |
| m9 | Email institucional – Validar email incluye "@tec.mx" | No | Agregar test específico para este caso |
| m10 | Retorno vacío – return {} directamente | Sí | - |

**Mutantes de matchesSearch**

| **Mutante** | **Tipo de error simulado** | **¿Falló la prueba?** | **¿Qué mejorar en la prueba?** |
| --- | --- | --- | --- |
| m1 | Comparación exacta proyecto – project.proyecto === search | Sí | - |
| m2 | Sin lowerCase en objetivo – project.objetivo\_ps.includes(search) | Sí | - |
| m3 | Retorno fijo falso – return false | No | Agregar test específico para este caso |
| m4 | Retorno fijo verdadero – return true | Sí | - |
| m5 | startsWith en proyecto – startsWith en vez de includes | Sí | - |
| m6 | endsWith en objetivo – endsWith en vez de includes | No | Agregar test específico para este caso |
| m7 | proyecto === '' – Comparar contra cadena vacía | Sí | - |
| m8 | objetivo === '' – Comparar contra cadena vacía | Sí | - |
| m9 | Negación de search – return !search | No | Agregar test específico para este caso |
| m10 | Igualdad exacta lower – project.proyecto.toLowerCase() === searchLower | Sí | - |

**Reflexión final**

* ¿Cuántos mutantes fueron detectados?:

Se detectaron 27 de 40 mutantes.

* ¿Qué patrones de debilidad encontraron en sus pruebas?

Durante la ejecución de los 40 mutantes, identificamos los siguientes patrones de debilidad en nuestras pruebas automatizadas:

* **Falta de cobertura para valores límite y casos borde:** Por ejemplo, no se detectaron mutantes que alteraban condiciones como >=, !==, o que usaban valores cercanos al umbral (como 60, 100, 180). Esto indica que no hay pruebas específicas para validar esos rangos.
* **Ausencia de validación para entradas inválidas o tipos incorrectos:** Mutaciones que introdujeron condiciones como typeof !== "number" no fueron detectadas, lo cual revela que nuestras pruebas no simulan entradas no numéricas o mal formateadas.
* **Dependencia en lógica exacta y no en comportamiento esperado:** Algunas pruebas no fallaron ante mutantes con retorno fijo (return true, return false), lo que indica que las pruebas no validan de forma específica los resultados esperados, sino que solo verifican que el código no falle.
* **Poca cobertura de ramas condicionales:** Mutantes que alteraron rutas específicas dentro de normalize, como cuando Resultado es "rejected" o "completed", pasaron sin ser detectados. Esto señala que nuestras pruebas no están cubriendo todas las posibles rutas del flujo de control.
* **Casos de prueba insuficientes para campos de formularios y validaciones:** Varias mutaciones dentro de validateForm no fueron detectadas porque no existen pruebas que validen específicamente campos como carrera\_completa, longitud de matrícula o validaciones alternativas de email.
* ¿Qué estrategias proponen para fortalecer los test?

Agregar pruebas de valores límite, cobertura de ramas, valores inválidos y aserciones más específicas.

# **Anexos:** Capturas y evidencia de ejecución de pruebas.

Link al Repositorio: <https://github.com/Fanilex/pruebas_mutantes>

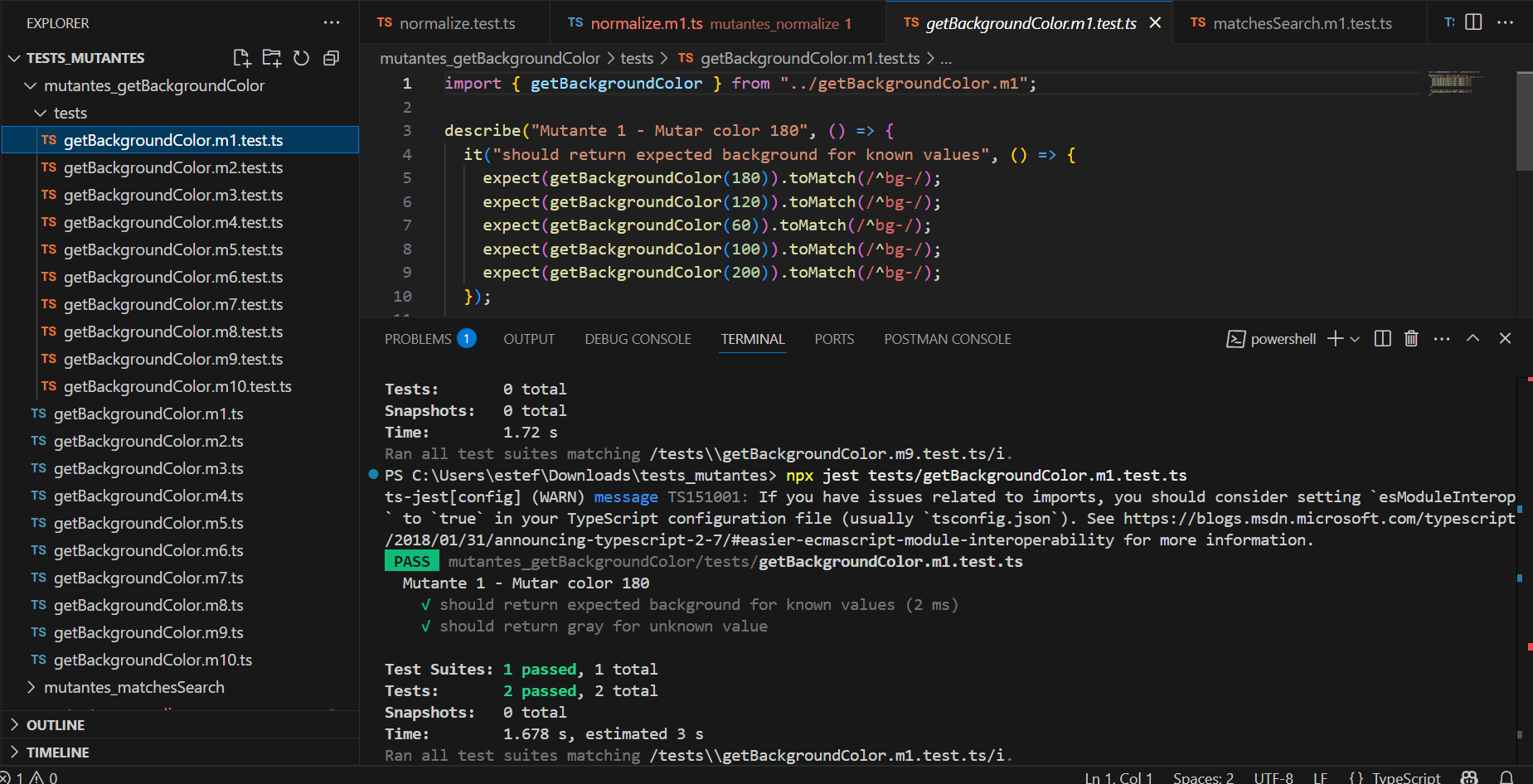


Figura 1. Captura de Test de getBackgroundColor donde pasó la prueba y no detectó el mutante.

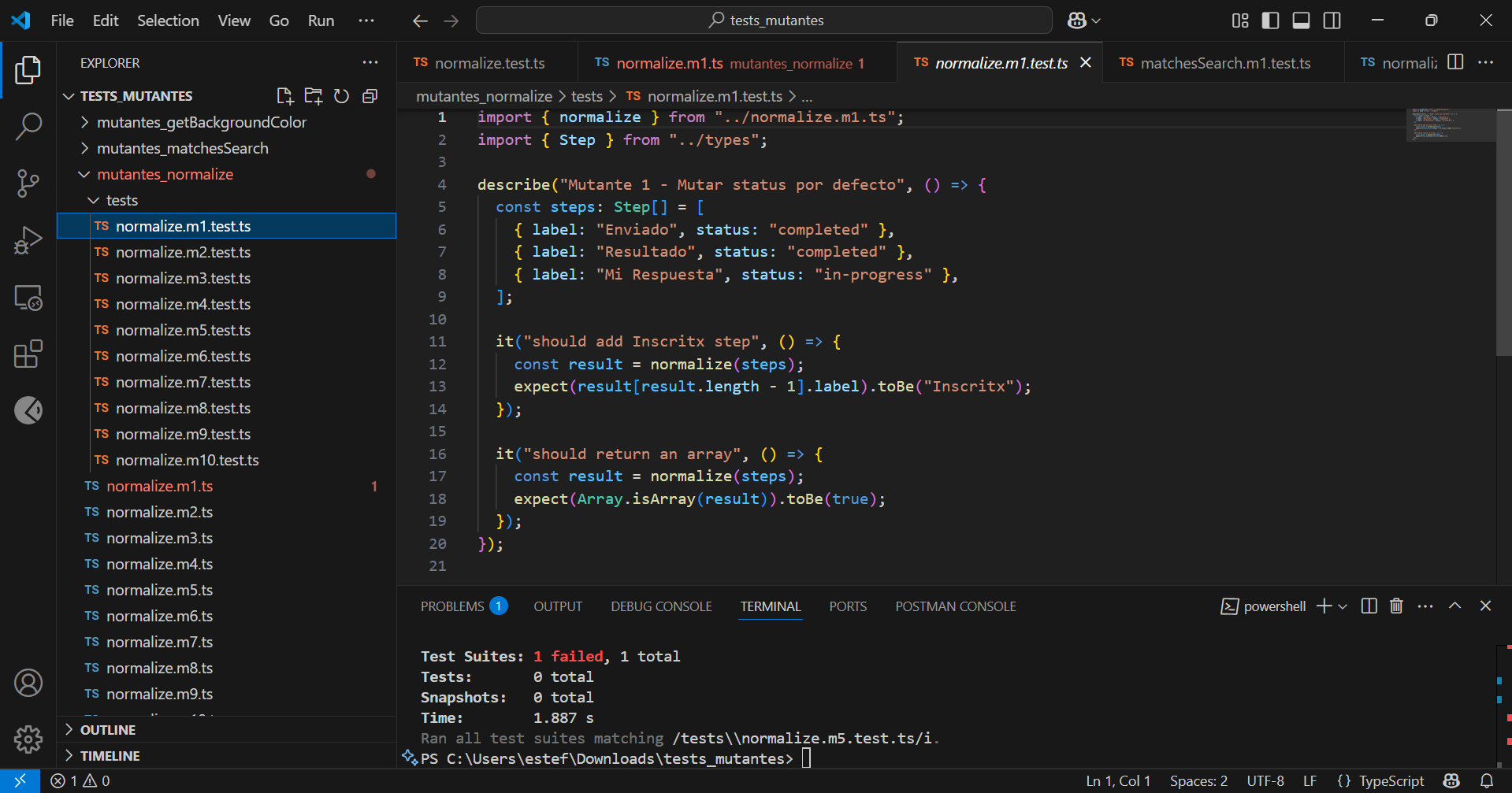


Figura 2. Captura de Test de Normalize

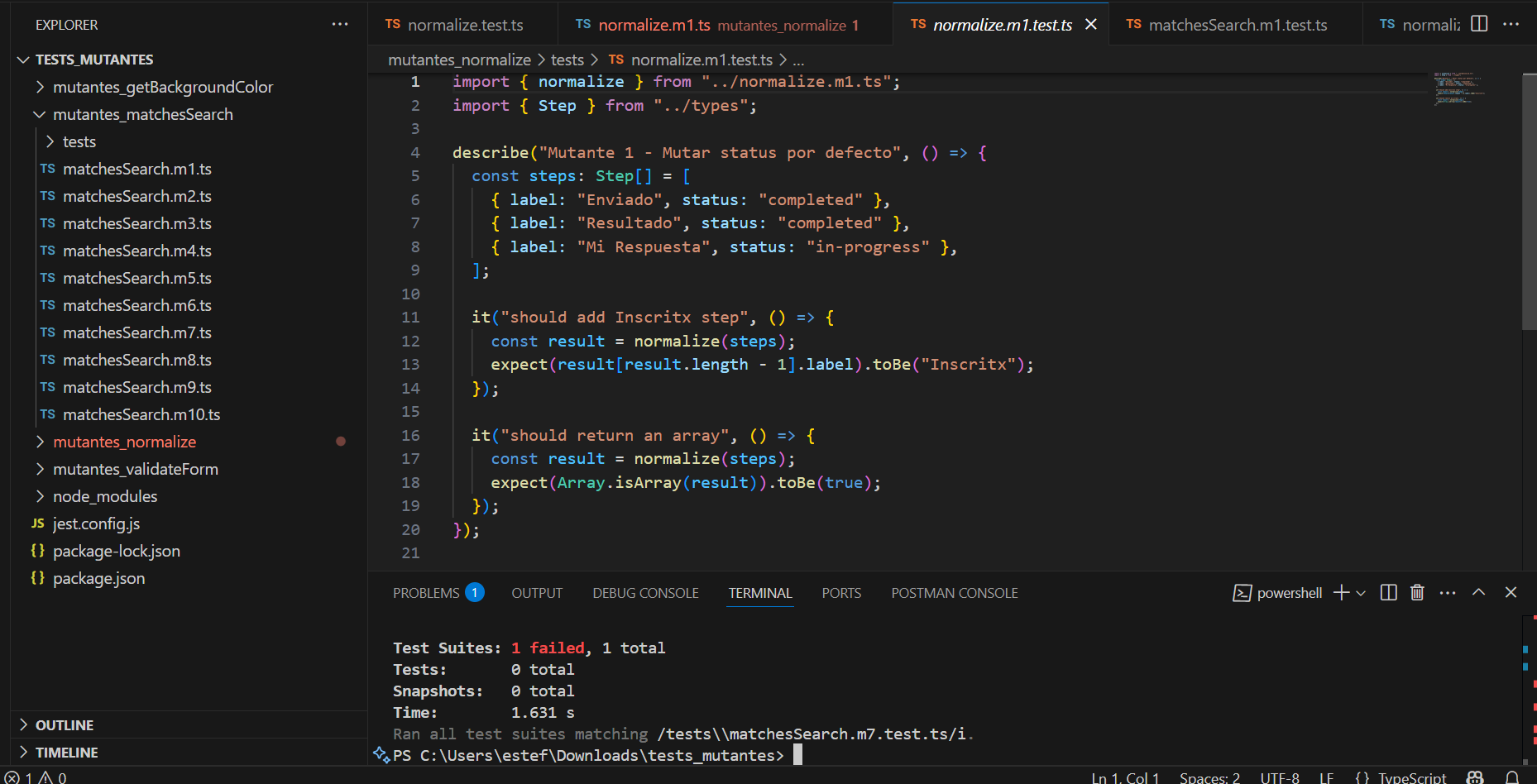


Figura 3. Captura de Test de Normalize

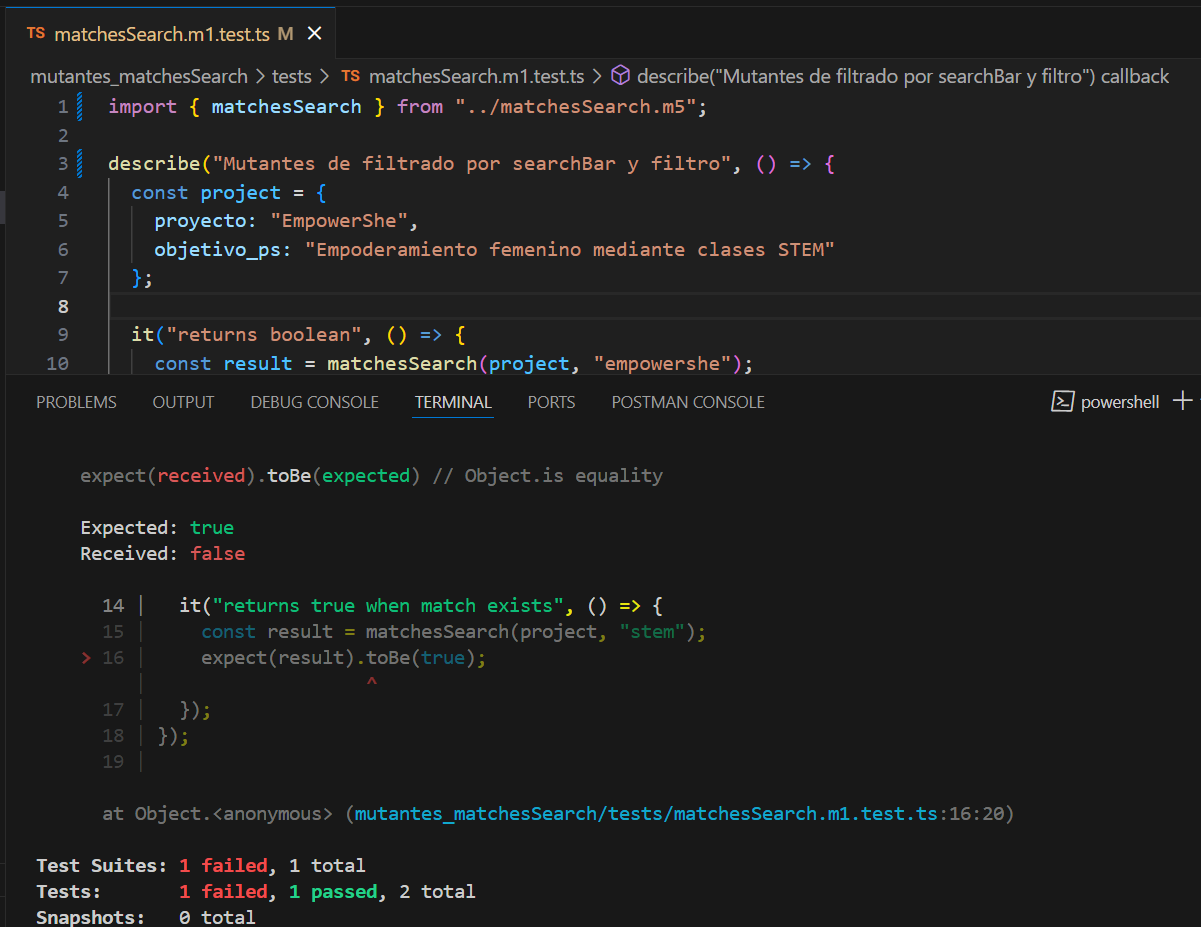


Figura 4. Captura de Test de matchesSearch