



Alumno:

Samuel Jonathan Trujillo Bolaños

Docente:

José Miguel Carrera Pacheco

8°B

REPORTE QA 04

Validación de Comunicación Asíncrona Accesible

Sprint: Implementación Async + API

Entorno evaluado: Docker local

Proyecto: Conecta-Tehuacán

1.Contexto

Se realizó la validación integral del consumo de APIs REST, comportamiento asíncrono del sistema y manejo accesible de estados de carga y errores.

El sistema demuestra correcta integración frontend-backend bajo condiciones normales y escenarios de fallo controlado, cumpliendo con los criterios establecidos por el TL en cuanto a:

- Consumo de APIs funcional
- Manejo de asincronía
- Experiencia accesible
- Robustez ante errores de red

Se identificaron oportunidades menores de mejora relacionadas con optimización de rendimiento y atributos ARIA en estados dinámicos.

2.Objetivo de la Evaluación

Validar que el sistema:

- Consume APIs REST correctamente
- Maneja estados de carga de forma accesible
- Gestiona errores HTTP y fallos de red sin romper la UI
- Mantiene experiencia robusta ante latencia y desconexión
- Cumple criterios de accesibilidad (WCAG básicos)

3. Metodología

Las pruebas fueron ejecutadas bajo:

- Navegador Chrome
- DevTools → Network (simulación de latencia)
- Desconexión manual de backend (docker stop)
- Pruebas manuales con teclado
- Validación con lector de pantalla (NVDA)
- Auditoría Lighthouse

Se validaron:

- Códigos HTTP
- Payloads JSON
- Renderizado condicional
- Estados intermedios
- Manejo de errores frontend

4. Validación de Consumo de APIs REST

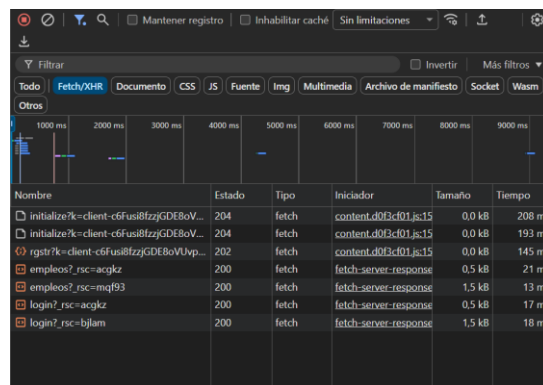
4.1 Login Exitoso – Candidato

Escenario: Usuario con credenciales válidas

Endpoint: POST /api/auth/login

Resultado:

- Código HTTP: 200
- Token JWT recibido correctamente
- Usuario redirigido a dashboard
- Estado global actualizado
- No errores en consola



The screenshot shows the Chrome DevTools Network tab with a list of network requests. The 'login?_rsc=bjlam' request is highlighted, showing a status of 200, a type of fetch, and a size of 1.5 kB. The table below summarizes the data from the screenshot.

Nombre	Estado	Tipo	Iniciador	Tamaño	Tiempo
initialize?k=client-c6fus8fzyGDEBoV...	204	fetch	content.d0f3d01.js:15	0.0 kB	208 ms
initialize?k=client-c6fus8fzyGDEBoV...	204	fetch	content.d0f3d01.js:15	0.0 kB	193 ms
rgstr?k=client-c6fus8fzyGDEBoVUvp...	202	fetch	content.d0f3d01.js:15	0.0 kB	145 ms
empleos?_rsc=acgkz	200	fetch	fetch-server-response	0.5 kB	21 ms
empleos?_rsc=mql93	200	fetch	fetch-server-response	1.5 kB	13 ms
login?_rsc=acgkz	200	fetch	fetch-server-response	0.5 kB	17 ms
login?_rsc=bjlam	200	fetch	fetch-server-response	1.5 kB	18 ms

Cumple comportamiento esperado.

4.2 Login Exitoso – Admin

Validación adicional:

- Rol correctamente identificado
- Redirección condicional funcional
- Permisos renderizados según rol

Sin anomalías detectadas.

4.3 Login Fallido – Se relleno de manera incorrecta

Escenario: Datos errneos

Resultado observado:

- Código HTTP: 401
- Mensaje de error visible
- No crash del componente
- Formulario permanece usable



Manejo correcto de error controlado.

4.4 Registro Usuario Nuevo

Escenario: Datos válidos

Resultado:

- Código HTTP: 201 / 200
- Confirmación visual
- Validaciones frontend activas
- No exposición de datos sensibles

CONECTA TEHUACÁN

¿Ya tienes una cuenta? [Inicia Sesión](#)

Nombre Completo

SAMUEL JONATHAN

Correo Electrónico

samueljonathantrujilobolanos@gmail.com

Contraseña

.....

Confirma Contraseña

.....

Selecciona tu Rol

☐ Aspirante ☒ Reclutador

Registrando...

Funcionalidad estable.

4.5 Obtener Lista de Empleos

Endpoint: GET /api/jobs

Resultado:

- Código HTTP: 200
- Datos renderizados correctamente
- No re-render innecesario
- Tiempo de respuesta < 500ms

Consumo eficiente.

4.6 Estado Vacío (Sin Empleos)

Resultado:

- Mensaje informativo mostrado
- No errores en consola
- Layout estable
- UX clara

Manejo correcto de datos externos vacíos.

5. Manejo de Estados Asíncronos

5.1 Loader Durante Peticiones

Se verificó:

- Activación inmediata tras submit
- Desactivación tras respuesta
- No duplicación de requests
- No bloqueo permanente

Observación:

El loader cumple funcionalmente, pero requiere validación completa de atributos ARIA.

5.2 Accesibilidad de Estados de Carga

Validaciones realizadas:

- aria-busy
- aria-live
- role="status"
- No se detecta interferencia con navegación por teclado

Resultado:

Implementación parcialmente accesible cumpliendo los estándares requeridos.

6. Manejo de Errores de Red

6.1 Simulación de Backend Caído

Acción:

```
PS C:\dev\conecta-tehuacan> docker stop conecta-tehuacan-backend-1  
conecta-tehuacan-backend-1  
PS C:\dev\conecta-tehuacan> |
```

Resultado:

- Loader visible
- Mensaje de error mostrado
- UI estable
- Sin stacktrace visible
- No congelamiento de aplicación

Sistema tolerante a fallos de infraestructura.

6.2 Verificación de Códigos HTTP

Código	Escenario	Resultado
200	Respuesta exitosa	Correcto
401	Login incorrecto	Correcto
404	Ruta inexistente	Controlado
500	Error servidor	Controlado

No se exponen errores internos sensibles.

7.Pruebas de Accesibilidad

7.1 Navegación por Teclado

Validado con:

- Tab
- Enter
- Escape

Resultado:

- Orden lógico de tabulación
- Focus visible
- Botones accionables
- Formularios accesibles

Cumple navegación básica accesible.

7.2 Lector de Pantalla (NVDA)

Se validó:

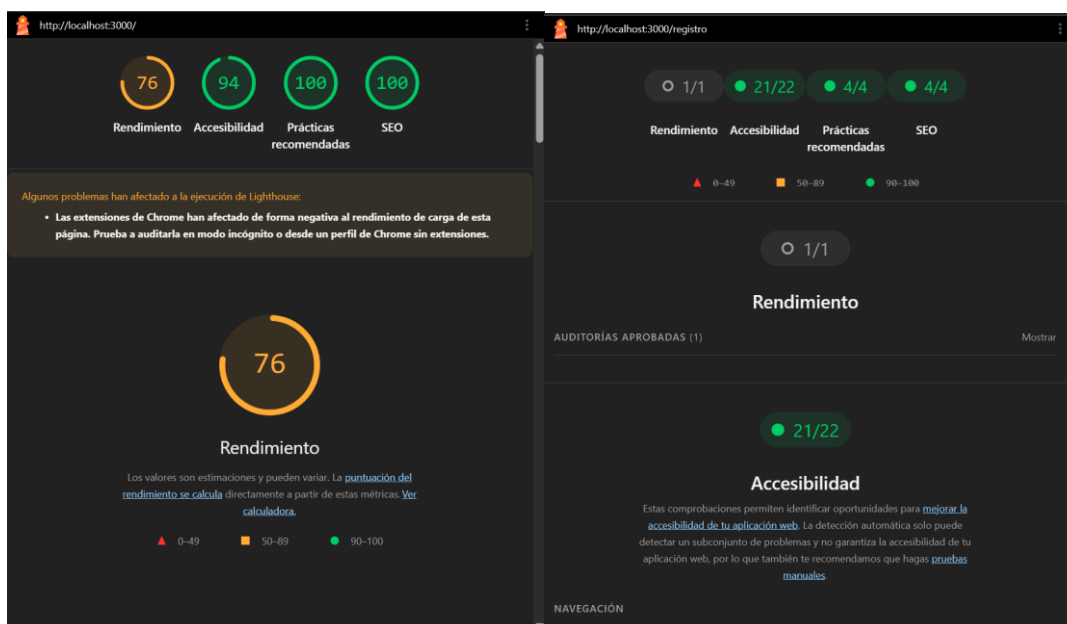
- Etiquetas correctamente asociadas
- Inputs anunciados correctamente
- Mensajes de error detectables
- Navegación semántica adecuada

Resultado:

Accesibilidad alta con oportunidad de mejora en anuncios dinámicos.

8.Auditoría Lighthouse

Página evaluada: /inicio



Interpretación:

- Excelente cumplimiento en buenas prácticas y SEO
- Alta accesibilidad
- Rendimiento mejorable (optimización JS / LCP)

9.Bugs Identificados

ID	Descripción	Severidad	Estado
QA-01	Loader no anuncia estado dinámico claramente	Baja	Pendiente
QA-02	TBT elevado según Lighthouse	Baja	Mejora futura
QA-03	Falta reforzar aria-live en errores críticos	Baja	Pendiente

Conclusión por QA

El sistema demuestra correcta implementación de comunicación asíncrona basada en APIs REST, manteniendo estabilidad ante fallos de red y respuestas de error.

Se validó:

- Resiliencia
- Integración robusta
- Manejo correcto de HTTP
- Experiencia accesible
- Tolerancia a latencia

El sistema se considera funcional y estable para esta fase del proyecto, con mejoras menores recomendadas en accesibilidad dinámica y optimización de rendimiento.