BANCO DE DESARROLLO RURAL
GERENCIA DE TECNOLOGÍA Y COMUNICACIONES
DIRECCIÓN DE CONTROL DE CALIDAD

EVALUACIÓN TESTER I

El objetivo de esta evaluación es medir ciertas competencias, el nivel de sus conocimientos y habilidad para aplicarlos en distintos ámbitos, así como su capacidad de analizar y relacionar conceptos. La evaluación está elaborada para candidatos al puesto de Tester I.

Instrucciones: Lea la evaluación detenidamente y resuelva cada inciso.

NOMBRE: Jerber Estuardo Cinto Rustrían . FECHA: 06/10/2023 .

I. **Conocimiento de seguridad y gestión de calidad** (15 puntos) Responda con sus propias palabras las siguientes preguntas:

A. ¿Qué es un fallo y de un ejemplo de fallo?
Un fallo en la seguridad de un desarrollo es aquello en donde queda expuesto o es vulnerable el proyecto, con ellos es fácil para el usuario poder acceder y extraer información valiosa.
Un ejemplo de ello es cuando el desarrollador no oculta el código fuente en el navegador haciendo que el usuario pueda acceder a esta información utilizando F12,

B. ¿Cuáles son las diferencias entre los tipos de pruebas de caja blanca y caja negra (Proporcione un ejemplo de cada uno)?
Caja blanca: son pruebas que se realizan a nivel backend o en la lógica de la programación
Ejemplo:
Un ejemplo de ello es el correcto funcionamiento que realice una llamada a la base de datos enviando o recibiendo peticiones a través del modelo y/o controlador

Caja negra: son pruebas que se realizan a nivel del frontend
Ejemplo:
Un ejemplo es cuando se valida que la entrada de texto de un formulario que acepte caracteres específicos y corrija o envíe una alerta si estos son incorrectos

C. Describe las etapas del ciclo de vida de las pruebas de software (Proporcione un ejemplo las cuales describan algunas de las etapas del ciclo de prueba). D. En términos de calidad de software, ¿Qué es el plan de testeo?

Existen 5 etapas en el ciclo de vida de software los cuales son:

Análisis:
lleva a cabo estrategias, especificaciones, pasos y aspectos para su correcto desarrollo

Ejemplo:

Se proporciona el problema o proyecto, para evaluar y analizar como añadir al desarrollo un nuevo modelo para las ventas, que el gerente pueda agregar un descuento a las personas que consuman mas de Q3,000.00

Diseño:
se implementan los documentos de diseño como, diagramas de flujo, diagramas de especificaciones o requerimientos, diagramas de clases entre otros

Ejemplo:

Se muestra los planos o diagramas para mostrar al cliente y desarrolladores el nuevo módulo de ventas añadiendo detalles a cada funcionalidad

Codificación:
Es cuando se toman las acciones para programar

Ejemplo:

Se acciona tomando un lenguaje de programación y se empieza a codificar basándose en los diseños

Pruebas:
se valida la programación y su correcto funcionamiento

Ejemplo:

Valida el QA que la nueva implementación del desarrollo no tenga fallas y el módulo de ventas que sea el correcto, sin tener conflicto con otras clases

Mantenimiento: Da paso como concluido el desarrollo e inicio a las declaraciones y actualizaciones del software del sistema

Ejemplo:

Cuando el gerente requiera un nuevo modulo a la parte de ventas o se encuentra el error que no todos los clientes les generen en descuento, eso da paso a el mantenimiento y actualización

E. ¿Qué elementos tomarías para crear una estrategia de pruebas?

1. Identificar qué tipo de pruebas son efectivas para cada ocasión como ejemplo las pruebas de testeo, caja blanca, caja negra.
2. Definir cuál es la finalizad de la prueba
3. Investigar que tipos de herramientas son efectivas para ayudar a los tipos de pruebas
4. Crear una estrategia y estimación de tiempo para tener el alcance y definir la conclusión de la misma

II. Casos de usos (25 puntos) Planteé su plan de ataque:

A. Encuentras un fallo en un proyecto de producción, ¿Cómo aseguras que el fallo sea resuelto?

- Indicar bajo que circunstancias se generó el error, es decir que es lo que el usuario estaba haciendo en el momento antes de generar el error
- Indicar al equipo de trabajo encargado sobre el tipo de error
- Documentar y describir a detalles sobre la falla
- Validar que tipo de prioridad o alerta genera el tipo de falla provocado
- Validar con el equipo de trabajo sobre las ultimas pruebas antes de entrar a producción
- Realizar un seguimiento sobre la falla brindando soluciones para la resolución del caso
- Realizar pruebas y garantizar el que el fallo ha sido corregido
- En el documento del proyecto registrar los detalles sobre la solución y sus modificaciones

B. El servidor de correo del proveedor está fallando, ¿Qué tipos de pruebas efectuarías para asegurar que el fallo sea del lado de proveedor?

- Indicar cuando surgió el error
- Comprobar conexión con el servidor
- Realizar pruebas con distintos tipos de redes
- Medir el tiempo de respuesta entre la petición del servidor
- Indicarle al proveedor el estado actúa del servidor,
- De no tener respuesta consultar con el proveedor y realizar pruebas con el técnico para la resolución del caso

C. Los clientes finales no tienen la idea de cómo validar el proyecto, ¿Qué harías como testar para obtener la aceptación del cliente final?

- Convocar a reuniones a los clientes sobre el avance junto con las pruebas que se tienen hasta el momento y explicar el funcionamiento
- Detallar en la documentación del proyecto como manuales de usuario
- En las reuniones hacer participe al usuario con las pruebas
- Establecer comunicación con el cliente, realizar retroalimentación y capacitaciones

D. El proyecto funciona de acuerdo con la lógica del cliente, Sin embargo, al tener un gran número de usuarios se observa una lentitud en la aplicación ¿Que tipos de pruebas efectuaría para visualizar este problema?

- Realizar pruebas de rendimiento y validar hasta cuantos usuarios es el máximo que puede soportar el proyecto
- Validar que las peticiones sean a nivel del frontend del sistema
- Verificar los recursos del servidor

E. Como tester observa que una aplicación tiene varias vulnerabilidades y riesgos, por lo tanto, ¿qué tipos de pruebas recomendarías para disminuir las vulnerabilidades y riesgos de esa aplicación?

- Validar si se pueda navegar en el sistema sin haber sido autenticado
- Validar que el usuario pueda ingresar a otras rutas sin ser autorizado
- Verificar que no se tengan acceso al código fuente en el navegador a través de las herramientas de desarrolladores
- Verificar que la ruta no se coloque información sensible para la empresa
- Validar la encriptación de datos

III. **Prueba técnica** (60 puntos)

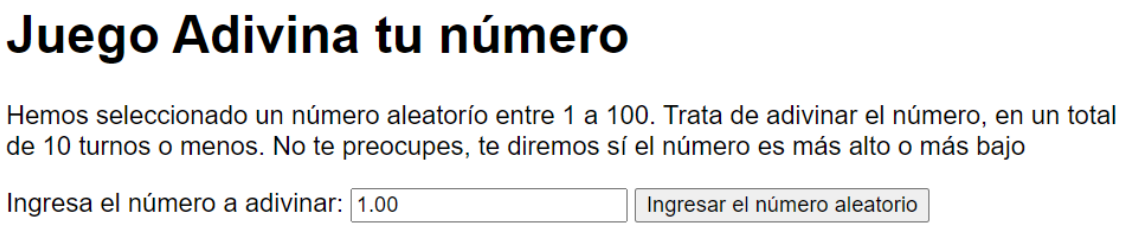
Visite el repositorio de GITHUB, TEST-QA-BANRURAL, y sigue todas las instrucciones del archivo README.md, posteriormente adjunte la URL de su repositorio, la cual contiene la solución del problema de la prueba técnica.

Prueba en el desarrollo "**Adivina tu número**"

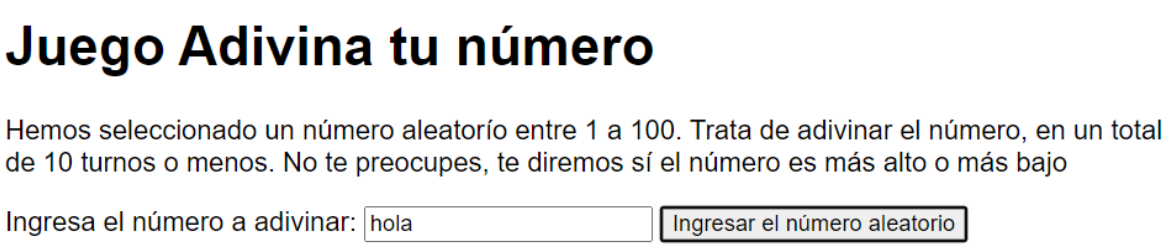
Módulo de pruebas caja negra:

En el siguiente análisis se realiza el siguiente testeo sobre el funcionamiento en la vista del usuario:

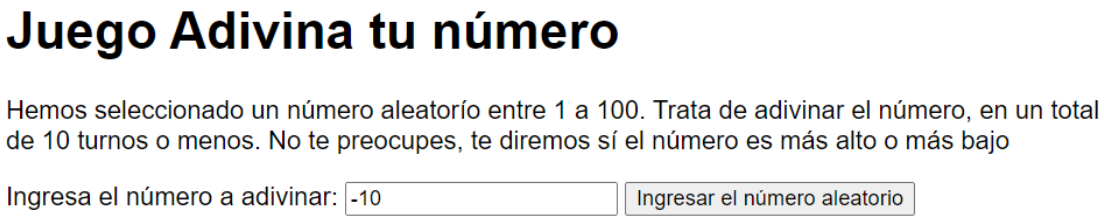
Se observa que el diseño de la página es muy simple, como corrección se debe de realizar los cambios necesarios como agregar un submenú que describa a detalle reglas del juego y un diseño que capture la atención del usuario



Se valida en el sistema que el campo acepta números reales y no muestra mensaje de error al accionar el botón



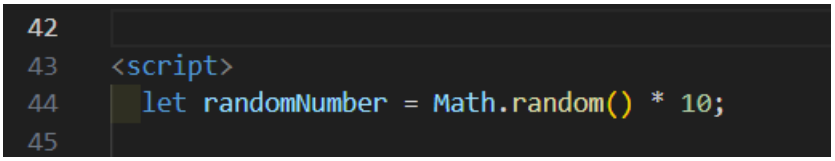
Se valida en el sistema acepta valores de tipo texto y no muestra mensaje de error al accionar el botón



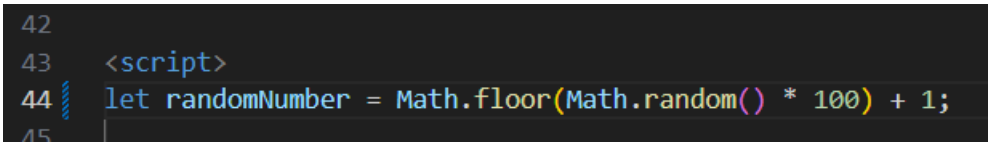
Acepta valores iguales o menores a 0

Análisis de caja blanca:

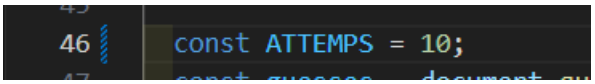
El numero que se genera aleatoriamente es incorrecto ya que este solo genera de 0 a 10



Se corrige el código, tomando el 1 como inicio, haciendo que genere un numero de 1 a 100



Se corrige la línea 46, permitiendo 10 intentos al usuario



Existe un error, el cual se escribe el nombre del evento **addEventListener** de forma incorrecta

Corrección:

```
}
guessSubmit.addeventListener('click', checkGuess);
                              abc addeventListener
                              addEventListener
function setGameOver() {
  guessField.disabled = true;
```

```
87    guessSubmit.addEventListener('click', checkGuess);
88
89    function setGameOver() {
90      guessField.disabled = true;
91      guessSubmit.disabled = true;
92      resetButton = document.createElement('button');
93      resetButton.textContent = 'Comienza un nuevo juego';
94      document.body.appendChild(resetButton);
95      resetButton.addEventListener('click', resetGame);
96    }
```

Se verifica que el llamado a la clase lowOrHi línea 49, no contiene la llamada como corresponde es decir el formato adecuado debe de ser un (.) seguido del nombre de la clase

```
43  <script>
44  let randomNumber = Math.floor(Math.random() * 100) + 1;
45
46    const ATTEMPS = 5;
47    const guesses = document.querySelector('.guesses');
48    const lastResult = document.querySelector('.lastResult');
49    const lowOrHi = document.querySelector('lowOrHi');
50    const guessSubmit = document.querySelector('.guessSubmit');
51    const guessField = document.querySelector('.guessField');
52
53    let guessCount = 1;
54    let resetButton;
55
56    function checkGuess() {
57
```

Corrección:

```
let randomNumber = Math.floor(Math.random() * 100) + 1;

  const ATTEMPS = 5;
  const guesses = document.querySelector('.guesses');
  const lastResult = document.querySelector('.lastResult');
  const lowOrHi = document.querySelector('.lowOrHi');
  const guessSubmit = document.querySelector('.guessSubmit');
  const guessField = document.querySelector('.guessField');

  let guessCount = 1;
  let resetButton;

  function checkGuess() {

```

Se detecta un error en la línea 65 y 70 del código fuente, los cuales no corresponden a la forma de validar si el cliente acertó o perdió en el juego, lo que genera ese tipo de confusión, adicional

- se corrige cuando el valor dado por el usuario, si es mayor o menor que el mensaje de respuesta lo devuelva en color negro
- en caso de que sobrepase los 10 intentos que devuelva en color rojo
- y en caso de que haya acertado devuelva el en color verde

```
    if(userGuess === randomNumber) {
      lastResult.textContent = '!!!Pérdistes!!!';
      lastResult.style.backgroundColor = 'black';
      lowOrHi.textContent = '';
      setGameOver();
    } else if(guessCount === ATTEMPS) {
      lastResult.textContent = 'Felicitaciones! adivinaste el número!';
      lastResult.style.backgroundColor = 'red';
      setGameOver();
    } else {
      lastResult.textContent = 'Incorrecto! ';
      lastResult.style.backgroundColor = 'green';
      if(userGuess < randomNumber) {
        lowOrHi.textContent = 'El número es mayor!';
      } else if(userGuess > randomNumber) {
        lowOrHi.textContent = 'El número es menor!';
      }
    }

```

Corrección:

```
69
70      if(userGuess === randomNumber) {
71        lastResult.textContent = '¡Felicitaciones! adivinaste el número!';
72        lastResult.style.backgroundColor = 'green';
73        lowOrHi.textContent = '';
74        setGameOver();
75      } else if(guessCount === ATTEMPS) {
76        lastResult.textContent = '!!!Pérdistes!!!';
77        lastResult.style.backgroundColor = 'red';
78        setGameOver();
79      } else {
80        lastResult.textContent = 'Incorrecto! ';
81        lastResult.style.backgroundColor = 'black';
82        if(userGuess < randomNumber) {
83          lowOrHi.textContent = 'El número es mayor!';
84        } else if(userGuess > randomNumber) {
85          lowOrHi.textContent = 'El número es menor!';
86        }
```

Se detecta un error de funcionamiento en el comienzo de la línea 56 que representa la funcionalidad para capturar el ultimo numero ingresado, el cual solo iguala al numero ingresado por el cliente

```
56    function checkGuess() {
57
58      let userGuess = guessField.value;
59      if(guessCount === 1) {
60        guesses.textContent = 'Número aleatorio anterior: ';
61      }
62      guesses.textContent += userGuess + ' ';
```

Se corrige la función para validar cuantos intento ha realizado el cliente y en caso de que no cumpla con los valores enteros entre 1 y 100 muestre una alerta, no se toma en cuenta y pueda realizar otro intento

```
58    function checkGuess() {
59    let userGuess = parseFloat(guessField.value);
60
61    if (!Number.isInteger(userGuess) || userGuess < 1 || userGuess > 100) {
62      alert('Ingresa un número entero válido entre 1 y 100.');
63      return;
64    }
65    if (guessCount === 1) {
66      guesses.textContent = 'Número aleatorio anterior: ';
67    }
68    guesses.textContent += userGuess + ' ';
```