1. OBJETIVO DEL PLAN DE CALIDAD

El presente Plan de Calidad tiene como objetivo definir la gestión de los procesos de calidad del proyecto “Lobo Empleo” durante todo su ciclo de vida. Este documento establece los estándares, responsabilidades, métricas, procesos y mecanismos de control que se aplicarán para garantizar que la plataforma:

- Valide correctamente los registros de estudiantes.
- Presente una vista gráfica clara, usable y accesible para el estudiante.
- Funcione de forma estable y eficiente de acuerdo con los requerimientos definidos en la materia de Herramientas Web.

2. ALCANCE DEL PLAN

El Plan de Calidad aplica a:

- Desarrollo front-end con React y Tailwind CSS.
- Desarrollo back-end con Node.js y PHP, incluyendo la API que conecta la base de datos con React.
- Diseño de la vista de estudiantes.
- Implementación del servicio de validación de estudiantes.
- Pruebas de funcionalidad del sistema.
- Documentación técnica y académica requerida para la evaluación del proyecto.

3. ESTÁNDARES DE CALIDAD

Los siguientes estándares se aplican de manera explícita al proyecto *Lobo Empleo*. Cada estándar se explica a detalle, indicando qué es, qué regula, por qué se usa y cómo se aplica a las tecnologías del proyecto: React, Tailwind CSS, PHP, Node.js, JavaScript y la API de conexión con la base de datos.

3.1 Estándar W3C (World Wide Web Consortium)

¿Qué es?

El W3C es el organismo internacional que define las reglas oficiales para la construcción de sitios web. Su objetivo es asegurar que las páginas y aplicaciones web sean compatibles, accesibles y funcionen correctamente en todos los navegadores.

¿Qué regula?

- Estructura semántica del HTML
- Buenas prácticas de CSS (código limpio, responsive, no redundante)
- Interacción y comportamiento con JavaScript
- Validación de formularios
- Adaptabilidad entre dispositivos

¿Por qué se usa en Lobo Empleo?

El proyecto utiliza **React**, lo que implica generar HTML dinámico. También usa **Tailwind CSS**, que debe integrarse correctamente a la estructura semántica. Cumplir con W3C evita errores visuales, problemas de responsividad y fallos en los navegadores.

Aplicación específica en el proyecto:

- Los componentes de React deben respetar etiquetas semánticas como <main>, <nav>, <header>, <section>.
- Los estilos en Tailwind CSS deben seguir principios de diseño limpio y responsive establecidos por el W3C.
- El código JavaScript usado en interacción debe evitar malas prácticas como inline scripting.
- El formulario de validación debe seguir las reglas de inputs correctos, validaciones básicas y mensajes descriptivos.

3.2 Estándar WCAG 2.1 (Web Content Accessibility Guidelines)

¿Qué es?

WCAG 2.1 es el estándar internacional para garantizar accesibilidad en sitios web para personas con discapacidad visual, auditiva, motriz o cognitiva.

¿Qué regula?

Organiza sus reglas bajo 4 principios fundamentales:

1. **Perceptible:** El usuario debe poder ver o escuchar el contenido.
2. **Operable:** La interfaz debe poder usarse por teclado y mouse.
3. **Comprensible:** El contenido debe ser fácil de entender.
4. **Robusto:** Compatible con tecnologías de asistencia (lectores de pantalla).

¿Por qué se usa en Lobo Empleo?

Porque la plataforma será utilizada por estudiantes, quienes pueden tener limitaciones visuales o de accesibilidad. Además, la vista del estudiante está siendo rediseñada, lo que exige cumplir con normas de accesibilidad moderna.

Aplicación específica en el proyecto:

- Contraste adecuado en Tailwind CSS (ej.: clases como text-gray-800 sobre bg-gray-100).
- Tamaños mínimos de fuente legible en React y Tailwind.
- Navegación coherente entre páginas con React Router.
- Botones con textos descriptivos, no solo íconos.

3.3 Estándar ISO 9241-210 (Diseño centrado en el usuario)

¿Qué es?

Es la norma internacional para el desarrollo de interfaces centradas en las necesidades y expectativas del usuario.

¿Qué regula?

- Cómo obtener retroalimentación real de usuarios.
- Cómo interpretar dicha retroalimentación para mejorar diseños.
- Cómo crear prototipos y evaluarlos antes del desarrollo final.
- Cómo validar que la interfaz sea intuitiva y funcional.

¿Por qué se usa en Lobo Empleo?

Porque el proyecto incluye encuestas, análisis, prototipos en Figma y pruebas con usuarios, que siguen exactamente el proceso de diseño centrado en el usuario.

Aplicación específica en el proyecto:

- TAREA 1 y 5 del rediseño: encuestas reales a estudiantes.
- TAREA 2: análisis de la vista actual según percepción del usuario.
- TAREA 3 y 4: prototipos siguiendo las preferencias del usuario.
- TAREA 5: validación del prototipo con 20 nuevos estudiantes.

Esto permite medir el KPI del "Índice de Satisfacción del Rediseño".

3.4 Estándar ISO/IEC 25010 (Calidad del Software)

¿Qué es?

Es el modelo de calidad más usado para evaluar productos de software. Define ocho características principales.

Características aplicadas al proyecto:

1. Funcionalidad

- El módulo de validación debe aceptar o rechazar registros según reglas definidas.
- La API debe responder correctamente a solicitudes de React.

2. Eficiencia del desempeño

- El tiempo de validación debe ser bajo (KPI: ≤ 5 segundos).
- Las respuestas del backend en Node.js/PHP deben ser rápidas.

3. Usabilidad

- La vista de estudiante debe ser clara, intuitiva y fácil de usar.
- Los formularios deben ser comprensibles.

4. Fiabilidad

- El sistema debe manejar errores sin fallar completamente.
- Las pruebas de funcionalidad deben demostrar estabilidad (≥ 85% aprobación).

5. Seguridad básica

- Los datos enviados entre React y la API deben manejarse con sanitización mínima.
- El backend debe validar inputs para evitar registros incorrectos.

3.5 Estándares REST para APIs (Node.js + PHP)

¿Qué es?

REST es el conjunto de buenas prácticas para diseñar APIs que comuniquen cliente y servidor usando rutas claras y datos estructurados.

¿Qué regula?

- Estructura de endpoints
- Métodos HTTP correctos: GET, POST, PUT, DELETE
- Respuestas JSON
- Manejo de errores
- Códigos de estado HTTP

Aplicación en el proyecto:

- La API en PHP/Node.js debe tener rutas como:
  - /api/validarEstudiante
  - /api/estudiantes
- La respuesta debe estar estructurada como JSON: { "mensaje": "validación exitosa", "status": 1 }Esto es crucial para garantizar consistencia entre la API y React.

3.6 Estándares de Pruebas de Software (Testing)

Descripción:
Conjunto de prácticas que definen cómo evaluar el funcionamiento real del sistema antes de su entrega.

¿Qué regula?

- Elaboración de casos de prueba
- Definición de escenarios y resultados esperados
- Ejecución con datos reales o simulados
- Registro formal de errores
- Priorización de correcciones
- Verificación final después de aplicar fixes

Aplicación en el proyecto:

- TAREA 1 del objetivo de pruebas: lista de casos de uso
- TAREA 2 y 5: pruebas con estudiantes reales
- TAREA 3: documentación de errores
- TAREA 4: correcciones en React + Backend

Permite evaluar formalmente el KPI: **“Porcentaje de funcionalidades aprobadas”**.

4. RESPONSABILIDADES EN LA GESTIÓN DE CALIDAD

- **Líder de proyecto:**
  - Aprueba este Plan de Calidad.
  - Supervisa el cumplimiento de estándares, procesos y cronograma.
  - Toma decisiones finales sobre la aceptación de entregables.

- **Jefe de desarrollo:**
  - Verifica la calidad técnica del código en React, Node.js y PHP.
  - Revisa la correcta implementación de la API y la integración con la base de datos.

- **Administrador de base de datos:**
  - Garantiza la integridad y consistencia de la información almacenada.
  - Proporciona datos para los KPI relacionados con validación de estudiantes.

- **Desarrollador front-end (React + Tailwind CSS):**
  - Implementa la vista de estudiantes conforme a los prototipos aprobados.
  - Aplica estándares W3C, WCAG 2.1 e ISO 9241-210 en la interfaz.

- **Desarrollador back-end (Node.js / PHP / API):**
  - Implementa el módulo de validación de estudiantes y las rutas de la API.
  - Optimiza el tiempo de respuesta y contribuye al KPI de tiempo de validación.

- **Diseñador UI/UX:**
  - Diseña prototipos de la vista de estudiantes.
  - Planifica y aplica encuestas de satisfacción y pruebas de usabilidad.

- **Tester:**
  - Diseña y ejecuta pruebas funcionales.
  - Registra errores y evidencia de resultados de los casos de prueba.

- **Documentador:**
  - Elabora y actualiza los documentos de calidad, incluyendo evidencias, reportes y versiones del sistema.

5. MÉTRICAS DE CALIDAD

La medición de la calidad se realiza mediante indicadores clave de desempeño (KPI), definidos en un documento independiente llamado **“Métricas de calidad”**. Para este proyecto se han definido, como mínimo, los siguientes:

- Tasa de validación correcta de estudiantes.
- Índice de satisfacción del rediseño de la vista de estudiantes.
- Tiempo promedio de validación por estudiante.
- Porcentaje de funcionalidades aprobadas en pruebas.

Cada KPI incluye: función, objetivos y cronogramas, línea base, unidad de medida, fuente de información, método de recolección, frecuencia y responsables de recolección y análisis.

6. PROCESOS DE CALIDAD

6.1 Proceso de calidad para el rediseño gráfico de la vista de estudiante

- Aplicar encuestas iniciales a estudiantes para evaluar la vista actual.
- Analizar la retroalimentación y definir qué elementos conservar, eliminar o mejorar.
- Investigar tipografías, colores e íconos adecuados.
- Elaborar prototipos en Figma siguiendo estándares W3C, WCAG e ISO 9241-210.
- Validar el prototipo con nuevos usuarios y registrar su satisfacción.
- Implementar el diseño aprobado en React + Tailwind CSS.

6.2 Proceso de calidad para la implementación del servicio de validación de estudiantes

- Investigar requisitos de validación (reglas y escenarios).
- Analizar librerías o servicios que faciliten la validación.
- Desarrollar el módulo de validación en Node.js/PHP y conectarlo a la base de datos mediante la API.
- Realizar pruebas internas con datos simulados para verificar que las reglas funcionen correctamente.
- Recoger retroalimentación de usuarios sobre la claridad del proceso de validación.
- Ajustar el módulo según resultados de pruebas y observaciones.

6.3 Proceso de calidad para las pruebas de funcionalidad del sistema

- Elaborar una lista de pruebas que incluya escenarios, casos de uso y resultados esperados.
- Ejecutar pruebas con datos simulados y con estudiantes reales.
- Documentar errores detectados, clasificarlos por prioridad e impacto.
- Corregir los errores en el código front-end y back-end.
- Reaplicar las pruebas hasta obtener el porcentaje de aprobación deseado.

7. CONTROL DE CALIDAD

7.1 Mecanismos de control

- Uso de **checklists de calidad** por fase (diseño, desarrollo, pruebas).
- Reuniones de revisión de avance según el cronograma del proyecto.
- Revisiones de código (code review) entre el jefe de desarrollo y los desarrolladores.
- Validación explícita del cumplimiento de estándares (W3C, WCAG, ISO 9241-210, ISO 25010).

7.2 Control mediante indicadores (KPI)

- Revisión periódica de los valores de los KPI.
- Comparación entre línea base, objetivo y resultados actuales.
- Registro de acciones correctivas cuando un KPI no alcanza la meta definida.

7.3 Documentación y evidencias

- Almacenamiento de encuestas, resultados de pruebas, capturas de pantalla y reportes en un repositorio organizado.
- Actualización del historial de versiones del proyecto (cambios en interfaz, validación y pruebas).

7.4 Aprobación final

Al concluir las fases de rediseño, validación y pruebas, el Líder de Proyecto revisará:

- Cumplimiento de los estándares definidos.
- Resultados finales de los KPI.
- Listas de verificación (checklists) completas.

Si los criterios de calidad se cumplen, se considerará terminada la versión académica del sistema “Lobo Empleo”.