<Aplicación para la Evaluación Docente>

Versión 1.0

Por José Ángel Ortiz Meraz, Gerardo Esteban Jurado Carrera.
Universidad Autónoma de Chihuahua

1. Descripción y Delimitación del Sistema

a. Nombre del Sistema:

Aplicación para la Evaluación Docente

b. Descripción y Delimitación del Sistema:

Esta es una aplicación tipo web/móvil diseñada para facilitar la evaluación de los docentes universitarios por parte de los estudiantes. Los estudiantes podrán realizar evaluaciones mediante cuestionarios que miden diferentes aspectos de la suficiencia docente. Los resultados estarán disponibles tanto para los docentes evaluados como para la administración de la universidad, con el fin de mantener y mejorar la calidad de la enseñanza.

c. Objetivo General:

Ofrecer un servicio para estudiantes y maestros universitarios que permita la consulta y realización de evaluaciones sobre la suficiencia de los docentes, contribuyendo a la mejora continua de la calidad educativa.

d. Objetivos Específicos:

1. Permitir a los estudiantes universitarios realizar evaluaciones de sus docentes a través de un cuestionario detallado.
2. Facilitar la consulta de las evaluaciones por parte de los docentes para fomentar su desarrollo profesional.
3. Proveer a la administración universitaria herramientas de análisis para mantener y mejorar los estándares de calidad educativa.
4. Garantizar la anonimidad y seguridad de las evaluaciones para promover una retroalimentación honesta y constructiva.

e. Descripción de Tipos de Usuarios:

1. **Estudiantes:** Usuarios que podrán acceder al sistema para realizar y consultar evaluaciones de sus docentes.
2. **Docentes:** Usuarios que podrán consultar los resultados de sus evaluaciones para obtener retroalimentación.
3. **Administradores Universitarios:** Usuarios encargados de gestionar el sistema, analizar los datos recolectados y tomar decisiones basadas en las evaluaciones.

f. Entorno Operativo del Sistema:

- **Plataformas:** Web y móvil (iOS y Android).
- **Tecnologías:**

    - Frontend: HTML5, CSS3, JavaScript (React para web, React Native para móvil).
    - Backend: Node.js con Express.
    - Base de Datos: PostgreSQL.
    - Seguridad: Autenticación basada en OAuth2, cifrado de datos sensibles.
- **Infraestructura:** Desplegado en la nube utilizando servicios como AWS o Azure para escalabilidad y fiabilidad.

2. Especificación de Requerimientos

a. Requerimientos Funcionales:

1. **Autenticación y Autorización:**
    - Registro y autenticación de usuarios.
    - Roles y permisos (Estudiante, Docente, Administrador).
2. **Gestión de Evaluaciones:**
    - Creación y edición de cuestionarios de evaluación.
    - Realización de evaluaciones por los estudiantes.
    - Consulta de evaluaciones por los docentes y administradores.
3. **Análisis y Reportes:**
    - Generación de reportes estadísticos sobre las evaluaciones.
    - Visualización de tendencias y áreas de mejora.
4. **Notificaciones:**
    - Notificación a los usuarios sobre nuevas evaluaciones disponibles.
    - Recordatorios y alertas importantes para los usuarios.
5. **Seguridad y Privacidad:**
    - Garantizar la anonimidad de las evaluaciones.
    - Protección de datos personales y resultados de evaluaciones.

b. Modelado del sistema
    i. Diagrama de clases
    ii. Diagrama de casos de uso
    iii. Diagrama de actividades

c. Requerimientos No Funcionales:

1. **Rendimiento:**
    - El sistema debe soportar al menos 1000 usuarios concurrentes.
    - Las respuestas a las evaluaciones deben procesarse en menos de 2 segundos.
2. **Escalabilidad:**
    - La arquitectura debe permitir el escalado horizontal para manejar incrementos en la carga de usuarios.

3. **Usabilidad:**
    - La interfaz debe ser intuitiva y fácil de usar tanto en dispositivos móviles como en navegadores web.
    - Debe proporcionar accesibilidad conforme a las pautas WCAG 2.1.
4. **Seguridad:**
    - Datos en tránsito y en reposo deben estar cifrados.
    - Implementación de controles de acceso estrictos según los roles de usuario.
    - Auditoría y registro de actividades de los usuarios.
5. **Mantenibilidad:**
    - El código debe seguir principios de diseño limpio y ser modular para facilitar el mantenimiento y la evolución del sistema.
6. **Disponibilidad:**
    - El sistema debe estar disponible al 99.9% del tiempo, exceptuando mantenimientos planificados.

3. **Arquitectura del Sistema/Aplicación.**

a. Layer/Tier's

El sistema sigue una arquitectura en capas donde se separan claramente el frontend y el backend:

- **Frontend (Cliente)**: Desarrollado utilizando Ionic con Angular, se encarga de la interfaz de usuario y la interacción con el usuario final.
- **Backend (Servidor)**: Implementado con Express.js, maneja la lógica de negocio, el acceso a la base de datos y la autenticación de usuarios.

c. Estructura modular del sistema

El sistema está estructurado en módulos que se comunican entre sí para realizar diferentes funciones:

- **Módulo de Autenticación**: Maneja el registro y la autenticación de usuarios, así como la gestión de roles y permisos.
- **Módulo de Evaluaciones**: Permite la creación, edición y realización de evaluaciones por parte de los estudiantes, y la consulta de resultados por parte de los docentes y administradores.
- **Módulo de Maestros**: Proporciona funcionalidades para ver, crear, actualizar y eliminar registros de maestros en la base de datos.

4. Base de datos
    a. Diagrama Conceptual (Entidad/Relación)
    b. Esquema Lógico de la Base de Datos

i. Especificación Tablas (Normalizadas hasta BCNF)
ii. Integridad de Datos (Constraints)
Claves primarias y foráneas: Cada tabla tiene una clave primaria única y se establecen las relaciones entre las tablas mediante claves foráneas.

5. **Lógica/Reglas del Negocio**

- Los maestros pueden ver las evaluaciones realizadas sobre ellos, pero no pueden editarlas.
- Los administradores tienen acceso completo para gestionar las evaluaciones y los registros de maestros.

6. **Descripción Interfaz de la aplicación**

La interfaz de la aplicación consta de una ventana principal con un menú lateral que permite navegar a las siguientes opciones:

- **Ver Maestros**: Permite a los usuarios ver una lista de maestros y sus detalles, pero sólo los administradores pueden realizar operaciones CRUD en los registros de maestros.
- **Evaluaciones**: Permite a los estudiantes realizar evaluaciones de maestros y a los docentes y administradores consultar los resultados de las evaluaciones.

7. **Descripción de reglas de seguridad(acceso/operación)**

Se implementan las siguientes reglas de seguridad:

- Control de acceso basado en roles para restringir las operaciones permitidas según el tipo de usuario.
- Cifrado de datos sensibles, como contraseñas, en la base de datos para proteger la información del usuario.

8. **Acceso a la Base de Datos mediante la Aplicación**

La aplicación se conecta a la base de datos Oracle 21c SQL utilizando consultas SQL parametrizadas para garantizar la seguridad y prevenir la inyección de SQL. Se utilizan controladores de Oracle específicos para Node.js para facilitar la comunicación con la base de datos.

Existen varios controladores de Oracle disponibles para Node.js que permiten la comunicación con la base de datos Oracle desde una aplicación Node.js. Algunos de los controladores populares incluyen:

- **`oracledb`**: Es un controlador Oracle específico para Node.js desarrollado por Oracle Corporation. Proporciona una API completa para interactuar con

una base de datos Oracle, incluyendo la ejecución de consultas SQL, la gestión de transacciones y el manejo de tipos de datos específicos de Oracle.

- **`node-oracledb`**: Es una implementación de `oracledb` para Node.js mantenida por la comunidad. Ofrece características similares a `oracledb` y es ampliamente utilizado en aplicaciones Node.js que requieren acceso a bases de datos Oracle.

9. **Conclusión**

En conclusión, la aplicación proporciona una plataforma eficiente y segura para que los estudiantes evalúen a los maestros universitarios, ayudando a mantener y mejorar la calidad educativa.