Proyecto #2 - NutriTEC

Instituto Tecnológico de Costa Rica
Área Académica Ingeniería en Computadores
Bases de Datos (CE3101)
II Semestre 2021
Valor 25%

Objetivo general

- ➔ Desarrollar una aplicación que permita manejar la descripción del caso expuesto.

Objetivos específicos

- ➔ Aplicar los conceptos del modelo conceptual y relacional.
- ➔ Crear una Base de Datos relacional en Microsoft SQL Server para que permita el almacenamiento de los datos.
- ➔ Crear una Base de Datos **no relacional** en MongoDB para que permita el almacenamiento de los datos de retroalimentación de los nutricionistas.
- ➔ Crear al menos 2 APIs (una para SQL Server y otra para MongoDB) para que controlen las funcionalidades y permitan la comunicación entre múltiples bases de datos.
- ➔ Crear una página Web para que exponga la funcionalidad al usuario.
- ➔ Usar herramientas como Angular, Bootstrap, HTML5, CSS y Reporting Services, Cristal Reports o similar para reportes.
- ➔ Crear un documento de instalación que permita el despliegue correcto de la aplicación en la nube.
- ➔ **Evaluar de forma objetiva, válida y precisa la solución planteada al problema complejo de ingeniería.**
- ➔ **Colaborar de forma activa en el equipo de trabajo para la realización del proyecto.**

Descripción del problema

NutriTec es una nueva plataforma para llevar un estilo de vida saludable específicamente en la nutrición de sus usuarios.
Los usuarios pueden ser independientes o estar asociados a un nutricionista. En caso de ser independientes solo deben ingresar una meta de consumo diario de calorías y empezar a realizar su registro diario
Si poseen un nutricionista ellos podrán realizarle un plan de alimentación a la medida para que su perdida ó aumento de peso sea lo más saludable posible.

Se ha determinado que son requeridas las siguientes vistas:

- **Vista Administrador:** Es la vista que usaran los administradores del sistema para aspectos de configuración de la plataforma.

- **Vista Cliente:** esta es la plataforma que permitirá a los clientes darse de alta, buscar productos, agregar metas, y registrar su consumo diario.
- **Vista Nutricionista:** esta es la plataforma que permitirá a los nutricionistas buscar y asociar clientes como sus pacientes y crearles y asignarles el plan alimenticio.

Requerimientos del Software

Vista Administrador.

- **Log In:** Se debe contar con un log in unificado que permita autenticar a la persona que está ingresando.

- **Aprobación de productos:** Esta opción permite al administrador aprobar si los datos ingresados por un cliente ó nutricionista son verdaderos (el proceso de verificación es manual).

- **Reporte de Cobro:** Desafortunadamente este es un proceso manual por lo que se solicita un reporte que facilite esta labor.
  Esta funcionalidad es prioritaria, permitirá al administrador saber el pago que debe realizarse a cada uno de los Nutricionistas debe estar agrupado por Tipo de Pago, posteriormente debe mostrar el correo electrónico, nombre completo del nutricionista, número de tarjeta y monto a cobrar.
  Esta funcionalidad debe ser implementada con un Store Procedure.
  - **¿Como se calcular el pago de cada uno de los nutricionistas?**
  - **Semanal:** Por cada uno de los pacientes que tiene asociados el nutricionista se le cobrará un dólar.
  - **Mensuales:** Por cada uno de los pacientes que tiene asociados el nutricionista se le cobrará un dólar y al total se le aplicará un 5% de descuento.
  - **Anual:** Por cada uno de los pacientes que tiene asociados el nutricionista se le cobrará un dólar y al total se le aplicará un 10% de descuento.

Vista Nutricionista.

- **Log In:** Se debe contar con una cuenta que permita la autenticación del usuario si posee una cuenta de lo contrario debe registrarse como nutricionista.

- **Registro:** Esta vista permitirá a los nutricionistas darse de alta. El nutricionista debe registrar su número de cédula, nombre y apellidos, código de nutricionista, edad, fecha de nacimiento, peso, IMC, dirección, foto, número de tarjeta de crédito (donde se realizará el cobro de la mensualidad), tipo de cobro (semanal, mensual, anual), correo electrónico y password (encriptado usando MD5).

- **Gestión de productos/platillos:** Esta opción permite al nutricionista agregar nuevos productos. De los productos se conoce un código de barras único, descripción, tamaño de la porción(g/ml), energía (Kcal), grasa (g), sodio(mg), carbohidratos(g), proteína (g), Vitaminas, Calcio(mg), Hierro(mg). Cualquier producto nuevo quedará a la espera de la aprobación del administrador para poder ser utilizado por toda la comunidad.

- **Búsqueda y Asociación de clientes como pacientes:** Esta funcionalidad permitirá al nutricionista buscar un cliente y asociarlo a su lista de pacientes.

- **Gestión de planes:** Esta opción permitirá a los nutricionistas gestionar los planes de alimentación, los cuales se componen de 5 tiempos de comida (Desayuno, Merienda Mañana, Almuerzo, Merienda Tarde y Cena) así como los productos/platillos a cada tiempo de comida que considere necesario todo esto acompañado de un nombre para el plan y el nutricionista que lo creo y totalizará la cantidad total de calorías.

- **Asignación de un plan a un paciente:** Esta funcionalidad le permite a un nutricionista asociar un plan a un paciente en específico. Con esto el paciente sabe cual debe ser su consumo máximo de calorías por tiempo de comida y total para su día.

- **Seguimiento Paciente:** El Sistema debe proporcionar la facilidad para que el nutricionista brinde una retroalimentación al paciente. En primera instancia el nutricionista puede ingresar al perfil de su paciente y revisar el registro diario del paciente y si así lo desea realiza una retroalimentación al paciente en un campo de prosa tipo foro. Al ser un foro y no limitar la extensión de la retroalimentación y sus respuestas esta información debe ser almacenada en una base de datos desarrollada en **MongoDB**.

Vista Cliente/Paciente.

- **Log In:** Se debe contar con una cuenta que permita la autenticación del usuario, si posee una cuenta de lo contrario debe registrarse. Si el paciente esta asociado con un nutricionista y tiene un plan de alimentación asignado justo después del log in se debe mostrar el plan asignado.

- **Registro:** Esta vista permitirá a los clientes darse de alta. El cliente debe registrar su nombre y apellidos, edad, fecha de nacimiento, peso, IMC, país donde reside, peso actual, medidas de: cintura, cuello, caderas, % de Musculo, % de grasa, consumo diario máximo de calorias, correo electrónico y password (encriptado usando MD5).

- **Registro de Medidas:** El sistema debe proveer funcionalidad de registrar para una fecha dada un registro de medidas: cintura, cuello, caderas, % de Musculo, % de grasa.

- **Registro diario de consumo:** Esta funcionalidad le permite al usuario registrar por cada uno de los tiempos de comida su consumo. Por ejemplo, puede seleccionar el desayuno y podrá

realizar una búsqueda de los alimentos por nombre o por código de barras identificado el alimento lo asigna a su registro diario de consumo.

- **Gestión de productos/platillos:** Esta opción permite al nutricionista agregar nuevos productos. De los productos se conoce un código de barras único, descripción, tamaño de la porción(g/ml), energía (Kcal), grasa (g), sodio(mg), carbohidratos(g), proteína (g), Vitaminas, Calcio(mg), Hierro(mg). Cualquier producto nuevo quedará a la espera de la aprobación del administrador para poder ser utilizado por toda la comunidad.

- **Gestión de Recetas:** Si a la hora de realizar el registro diario no existe el producto o platillo específico y este puede ser creado a partir de los productos ya existentes el sistema debe proveer la funcionalidad de crear una receta. Por ejemplo, El pinto para el desayuno no existe pero tenemos el arroz, frijoles y condimentos dentro de la base de datos productos-alimentos podríamos crear una receta llamada Pinto donde agregamos las porciones necesarias del arroz, frijoles y condimentos y el sistema debe totalizar las calorías, carbohidratos, azúcares, vitaminas, etc para la receta creada. Así el usuario puede agregar el pinto a su desayuno de ahora en adelante sin estar agregando el arroz y los frijoles por separado.

- **Reporte de Avance:** El sistema debe proveer un reporte que muestre los valores registrados para las medidas en un periodo de tiempo dado. Osea el usuario selecciona una Fecha Inicio y una Fecha Final y el sistema gráfica para todas las fechas dentro del periodo seleccionado las medidas que el usuario ha ido registrando.

App Móvil.

- **Log In:** Se debe contar con una cuenta que permita la autenticación del usuario.

- **La funcionalidad de la app móvil debe ser la expuesta en el cliente en la sección registro diario y Gestión de Recetas.**

El término gestionar corresponde a las opciones de insertar, editar, eliminar y consultar.

Para la revisión del proyecto:

- ➔ Aparte del procedimiento almacenado solicitado en la especificación, el profesor seleccionará tres adicionales para ser evaluados. En total se evaluarán cuatro procedimientos almacenados (el obligatorio y tres adicionales seleccionados durante la revisión).
- ➔ Deben implementarse al menos 3 triggers.
- ➔ Deben implementarse al menos 2 vistas.

Requerimientos no funcionales del sistema

- ➔ El Sistema debe ser una aplicación web (utilizando Angular, Bootstrap, HTML5, CSS).
- ➔ Las Bases de Datos deben estar en Microsoft SQL Server y MongoDB.
- ➔ **Toda la lógica de la Base de Datos debe ser implementada mediante Store Procedures, Vistas y Triggers.**
- ➔ La capa de servicios debe estar desarrollada en C# y debe ser desplegada en **Azure o AWS** al igual que la App web.
- ➔ El equipo de trabajo debe seleccionar a uno de sus miembros como único punto de contacto. Todas las comunicaciones y solicitudes deben ser a través de dicho punto de contacto.

Entregables

- ➔ Manual de Usuario.
- ➔ Documento de evidencia de la solución elaborada y planificación del trabajo.
- ➔ Se deberá documentar el código fuente.
- ➔ Evidencia de uso de un manejador de código (se recomienda Gitgub).
- ➔ Documento de instalación.
- ➔ Plan de Proyecto **(Anexo)**.
- ➔ Script de Base de Datos.
- ➔ Script de población de Base de Datos.
- ➔ Aplicación WEB.
- ➔ Web APIs.
- ➔ Minutas.

NOTA: Cada documento solicitado debe tener la estructura de un documento técnico (portada, índice de contenidos, introducción, entre otros).

Documentación evidencia de solución elaborada y planificación del trabajo

- ➔ Se deberá entregar un documento que contenga:
  - ◆ Modelo conceptual utilizando la notación de Chen.
  - ◆ Modelo relacional.
  - ◆ Descripción de las estructuras de datos desarrolladas (Tablas).
  - ◆ Descripción de los Store Procedures, Triggers y Vistas implemendos.
  - ◆ **Descripción detallada de la arquitectura desarrollada.**
  - ◆ Problemas conocidos: En esta sección se detalla cualquier problema que no se ha podido solucionar en el trabajo.
  - ◆ Problemas encontrados: descripción detallada, intentos de solución sin éxito, soluciones encontradas con su descripción detallada, recomendaciones, conclusiones y bibliografía consultada para este problema específico.
- ➔ Conclusiones del proyecto.
- ➔ Recomendaciones del proyecto.
- ➔ Bibliografía consultada en todo el proyecto.

Aspectos operativos y evaluación:

1. **Fecha de entrega:** De acuerdo con el cronograma del curso y lo establecido en el TEC Digital. Se establece el siguiente plan de entregas parciales:
   a. **Anexo**: 22/Oct/2021
   b. Resumen Ejecutivo Avance 1: 29/Oct/2021
   c. Resumen Ejecutivo Avance 2: 5/Nov/2021
   d. Funcionalidad completa: 12/Nov/2021
2. El proyecto tiene un valor de **25%** de la nota del curso. Un **5%** corresponde al anexo.
3. El trabajo es **en grupos de 4 personas**.
4. La implementación tendrá un valor de un 70% de la nota final, debe estar funcional. La defensa vale un 10% y la documentación externa un 20%.
5. Cumplir con los requerimientos especificados en la documentación no significa que se tienen todos los puntos, se evaluará que la documentación sea coherente, acorde al tamaño del proyecto y el trabajo realizado, no escatimen en documentación.
6. Cada grupo recibirá una nota en cada uno de los siguientes apartados Código, Defensa y Documentación.

7. El profesor no sólo evaluará la funcionalidad del proyecto, esto quiere decir que aunque el proyecto este 100% funcional esto no implica una nota de un 100, ya que se evaluarán aspectos de calidad de código, aplicación del paradigma **OOP**, uso de herramientas solicitadas, calidad de documentación interna y externa, trabajo en equipo.
8. No se revisarán funcionalidades parciales, ni funcionalidades no integradas.
9. Es responsabilidad de cada miembro del grupo conocer su código, el profesor puede preguntar a cualquier miembro del grupo que le explique alguna funcionalidad/porción de código.
10. Las citas de revisión oficiales serán determinadas por el profesor durante las lecciones o mediante algún medio electrónico.

11. Aún cuando el código y la documentación tienen sus notas por separado, se aplican las siguientes restricciones:
    a. Si no se entrega documentación, automáticamente se obtiene una nota de cero en el proyecto.
    b. Si no se utiliza un manejador de código se obtiene una nota de cero en el proyecto.
    c. Si el código y la documentación no se entregan en la fecha indicada se obtiene una nota de cero en el proyecto.
    d. Si el código no compila se obtendrá una nota de cero en el proyecto, por lo cual se recomienda realizar la defensa con un código funcional.
    e. Si el grupo no cuenta con los equipos necesarios para realizar la revisión y no avisó al profesor de esta situación obtendrá una nota de cero en el proyecto.
    f. El código debe ser desarrollado en C#, en caso contrario se obtendrá una nota de cero en el proyecto.
    g. Si el grupo no se presenta a la revisión/defensa se obtiene nota de cero en el proyecto. Si un estudiante no se presenta a la defensa y no cuenta con una justificación válida se le asignará al estudiante una nota de cero en el proyecto.
12. Cada grupo tendrá como máximo 50 minutos para exponer su trabajo al profesor y realizar la defensa de éste, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cual se recomienda tener todo listo antes de ingresar a la defensa y el profesor dispondrá de 10 minutos para dar la retroalimentación y la nota del proyecto.
13. Cada excepción o error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final del proyecto.
14. Cada grupo es responsable de llevar los equipos requeridos para la revisión.
15. Durante la revisión únicamente podrán participar los miembros del grupo, asistentes, otros profesores y el coordinador del área.
16. Las revisiones se realizan con los estudiantes matriculados en el curso, cualquier persona fuera de estos y los mencionados en el punto 15, no pueden participar en la revisión.

Referencias

AngularJS (2018-10-04). Recuperado de**:** **https://angularjs.io**

Bootstrap Themes & Templates (2018-10-04). Recuperado de**:** **https://wrapbootstrap.com/**

How to Write Doc Comments for the Javadoc Tool. (2018-10-04). Recuperado de: http://www.oracle.com/technetwork/articles/java/index-137868.html

C# Coding Conventions (C# Programming Guide). (2018-10-04). Recuperado de: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions