Tecnológico de Costa Rica

Customer Relationship Manager
Reporte de auditoría de código

Version 1.1

Índice

Información del documento **3**
- Descripción 3
- Identificación del documento 3
- Organización encargada 3
- Historial de cambios 3

Introducción **4**
- Alcance 4
- Referencias 4
- Glosario 4

Desviaciones con respecto al estándar **4**

Observaciones de la implementación **5**
Recomendaciones y acciones a tomar **6**

1. Información del documento

1.1. Descripción

El siguiente es un reporte que detalla el resultado de la revisión del código fuente del producto *Customer Relationship Manager*, referido de ahora en adelante como CRM.

Versión del software a revisar: 3.2.

1.2. Identificación del documento

CI102017-1

1.3. Organización encargada

Los siguientes recursos componen la organización encargada de llevar a cabo la inspección pertinente.

- Izcar Muñoz
- Isaac Campos
- Jeffrey Alvarado

1.4. Historial de cambios

Fecha	Versión	Descripción de los cambios	Autor
01/10/2017	1.0	Primera versión	Jeffrey Alvarado
30/10/2017	1.1	Se especifica la versión del software a evaluar	Jeffrey Alvarado

2. Introducción

2.1. Alcance

El propósito de este documento es proporcionar un informe resultante del proceso de reunión e inspección de código fuente llevado a cabo en el producto de software CRM. Durante este reporte, se llevará a cabo una revisión de la implementación de los primeros dos conjuntos de funciones implementados: Manejo de usuarios y Manejo de contactos.

2.2. Referencias

[DEP] Documento de estándar de programación

2.3. Glosario

Las siguientes abreviaciones son usadas en el presente documento:

CRM Customer Relationship Manager

DRY Don’t Repeat Yourself

3. Desviaciones con respecto al estándar

- El código presenta desviaciones principalmente con respecto al nombrado poco significativo de variables y archivos. En estas encontramos variables con nombres por defecto como *Button1* o *WebForm1* en los archivos *login.aspx* e *index.aspx*.
- También varios nombres presentan errores ortográficos.
- Igualmente maneja de manera inconsistente sus costumbres de nombrado, ya que a veces estos aparecen en español y otras en inglés como se muestra en la imagen:
- Hay archivos no utilizados dentro de la carpeta del proyecto, residuales de la plantilla utilizada en la carpeta *pages* y en la raíz del proyecto.
- Se presenta còdigo duplicado al inicio de cada página, ya que reescribe el encabezado de cada una, lo que viola el principio DRY.

- Segùn el estándar [DEP], la manera de concatenar cadenas de caracteres es con la función nativa *Append()*, la cual no es usada.
- Faltan comentarios que indiquen cómo se está llevando a cabo la validaciòn del formato de correo en el registro de usuarios y contactos.
- Se hacen operaciones con AND y OR sin paréntesis.

4. Observaciones de la implementación

- Falta validar la cantidad de filas recuperadas en la sección de ingreso al sistema. Esto para evitar problemas con inyecciones SQL.
- La función de ingreso al sistema refresca la interfaz si falla la validación de datos. Esto es innecesario y lleva a la pérdida de información.
- Muchas funciones generadas automáticamente por el IDE se mantienen en el código y no son utilizadas.
- El sistema verifica el número de teléfono solo en el *backend*, lo cual puede llegar a ser contraintuitivo, ya que permite el ingreso de datos que no son números, a pesar de que no los valide.
- Se repite código en algunas condiciones, por ejemplo en *contactoEmpresa.aspx*:

```
54: (nombreEmpresa.Equals(“”) || nombreEmpresa.Length == 0)
```

- La función de verificar el nombre de la empresa y la validez de la dirección hacen lo mismo.
- Algunas funciones son muy grandes y pueden simplificarse en muchas más pequeñas. Véase *verificarDatosEmpresa()* en el archivo *contactoEmpresa.aspx*.
- Mala implementación del registro de usuarios con la base de datos. El sistema crea dos conexiones distintas para realizar las transacciones, lo cual no es eficiente ni claro para el programador.
- Algunas interacciones con las consultas no son atómicos, lo que hace al sistema menos tolerante a fallos.
- Desacoplar código del cliente y del servidor. En algunas secciones se incluye código HTML en archivos de control.

- Se pueden encontrar nombres y títulos residuales del template utilizado

5. Recomendaciones y acciones a tomar

1. Cambiar los nombres autogenerados en los archivos *.aspx* por nombres más significativos.
2. Revisar y renombrar todas las variables y archivos que están en escritos inglés a nombres en español.
3. Corregir el nombre del archivo *conecion.cs* por *conexion.cs* así como las variables que hacen referencia a la conexión.
4. Eliminar la carpeta *pages* y todos los archivos *.html* del proyecto.
5. Apegarse al principio DRY, esto implica crear archivos de encabezado y pie de página que contenga el mismo código con necesario para las distintas páginas así como las referencias a las bibliotecas importadas.
6. Usar los métodos nativos del lenguaje para realizar acciones sobre cadenas de caracteres.
7. Asegurarse de validar la cantidad de filas recuperadas en la consulta que autoriza la entrada al sistema.
8. Eliminar o darle uso a las funciones autogenerada por el IDE.
9. Prohibir el ingreso de datos no válidos desde el cliente.
10. Dividir las funciones en otras más pequeñas cuando sea aplicable.
11. Cerrar la conexión con la base de datos una vez terminadas las consultas en las funciones que interactúan con la base de datos y comprimir la funcionalidad de la duplicidad de los objetos de conexión a un único objeto reutilizable.
12. Hacer *commit* de los datos de las consultas con las bases de datos inmediatamente antes de cerrar la conexión.
13. Mantener el código HTML en archivos aparte que se puedan consultar para hacer la respectiva inyección del código al cliente.