**LISTA DE CHEQUEO PARA EL DOCUMENTO DE ESPECIFICACIÓN DE REQUISITOS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Revisado por** | **Walter Renzo Muñoz Arellano** | **Cargo o rol** | | | | **Tester** | |
| **Fecha** | **22/10/2019** | **Sitio** | | | | **FISI** | |
| **Nombre del proyecto** | **Sistema de Biblioteca - FISI** | | | | | | |
| **Descripción: Sistema que facilita el préstamo e inventario de los libros que se encuentra en la biblioteca de la Facultad de Ingeniería de Sistemas e Informática.** | | | | | | | |
| **REVISIÓN DEL CONTENIDO GENERAL DEL DOCUMENTO** | | | | | | | |
| **ASPECTO O ELEMENTO A VERIFICAR** | | | **SI** | **NO** | **NO APLICA** | | **OBSERVACIONES** |
| 1. **Introducción** | | |  | **X** |  | | **Figura que se encuentra en el documento “Manual de Usuario” pero no figura en su índice** |
| 1.1. Propósito | | |  | **X** |  | | **Figura que se encuentra en el documento “Manual de Usuario” pero no figura en su índice** |
| 1.2. Ámbito o alcance | | |  | **X** |  | |  |
| 1.3. Definiciones, siglas y abreviaturas | | |  |  | **X** | | **No ha sido adjuntado** |
| 1.4. Referencias | | |  | **X** |  | | **No ha sido adjuntado** |
| 1.5. Apreciación Global del documento | | |  |  | **X** | |  |
|  | | | | | | | |
| **2. Descripción general** | | |  |  |  | |  |
| 2.1. Perspectiva del producto | | |  |  | **X** | |  |
| 2.2. Funciones del producto | | | **X** |  |  | | **La funciones se encuentra detalladas en el Manual de Usuario** |
| 2.3. Características de los usuarios | | |  | **X** |  | | **No se encontró el documentos con estas especificación.** |
| 2.4. Restricciones | | |  | **X** |  | | **Sin especificar** |
| 2.5. Suposiciones y dependencias | | |  | **X** |  | | **Sin especificar** |
| 2.6. Requisitos futuros | | |  | **X** |  | | **Sin especificar** |
|  | | | | | | | |
| **3. Requisitos Específicos** | | |  |  |  | |  |
| 3.1. Interfaces Externas | | | **X** |  |  | | **Se muestran en el documento de “Manual de Usuario”** |
| 3.1.1 Interfaces del Sistema | | |  |  | **X** | |  |
| 3.1.2 Interfaces del Usuario | | | **X** |  |  | | **Se muestran todas las acciones disponible para los usuarios.** |
| 3.1.3 Interfaces del Hardware | | | **X** |  |  | | **Se muestra el hardware mínimo para poder levantar el proyecto.** |
| 3.1.4. Interfaces del Software | | | **X** |  |  | | **Se muestra el hardware mínimo para poder levantar el proyecto.** |
| 3.1.5. Interfaces de Comunicaciones | | |  |  | **X** | |  |
| 3.2. Funciones | | | **X** |  |  | |  |
| 3.3. Requisitos de rendimiento | | |  |  | **X** | | **No se estableció** |
| 3.4. Requisitos de datos | | |  |  | **X** | | **No se estableció** |
| 3.5 Restricciones de diseño | | |  |  | **X** | | **No se estableció** |
| 3.6. Atributos del sistema | | |  |  | **X** | | **No se estableció** |
| 3.6.1. Funcionalidad | | |  | **X** |  | |  |
| 3.6.1.1. Adecuación | | |  | **X** |  | |  |
| 3.6.1.2. Exactitud | | |  | **X** |  | |  |
| 3.6.1.3. Interoperabilidad | | | **X** |  |  | | **Puede ser operado en diferentes sistemas Operativos** |
| 3.6.1.4. Seguridad de acceso | | | **X** |  |  | | **Se tiene un backend con el que se valida a los usuarios y sus acciones.** |
| 3.6.1.5. Cumplimiento funcional | | |  |  | **X** | | **No se adjuntan Historias de Usuario o Documento de Análisis** |
| 3.6.2. Fiabilidad | | |  | **X** |  | | **No paso según la prueba del SonarQube** |
| 3.6.2.1. Madurez | | |  |  | **X** | |  |
| 3.6.2.2. Tolerancia a fallos | | | **X** |  |  | | **Se encontró defectos en la aplicación pero que no afectan la funcionalidad** |
| 3.6.2.3. Capacidad de recuperación | | |  |  | **X** | | **No establecido** |
| 3.6.2.4. Cumplimiento de la fiabilidad | | |  | **X** |  | | **El SonarQube no aceptó su fiabilidad** |
| 3.6.3. Usabilidad | | | **X** |  |  | | **Propia experiencia** |
| 3.6.3.1. Capacidad para ser entendido | | | **X** |  |  | | **Propia experiencia** |
| 3.6.3.2. Capacidad para ser aprendido | | | **X** |  |  | | **Propia experiencia** |
| 3.6.3.3. Capacidad para ser operado | | | **X** |  |  | | **Propia experiencia** |
| 3.6.3.4. Capacidad de atracción | | |  | **X** |  | | **Falta de más pruebas de usuario** |
| 3.6.3.5. Cumplimiento de la usabilidad | | | **X** |  |  | |  |
| 3.6.4. Eficiencia | | |  | **X** |  | |  |
| 3.6.4.1. Comportamiento temporal | | |  |  |  | | **No establecido** |
| 3.6.4.2. Utilización de recursos | | | **X** |  |  | | **Se colocó una cantidad de libros en base de datos aceptable para pruebas reales.** |
| 3.6.4.3. Cumplimiento de la eficiencia | | |  | **X** |  | |  |
| 3.6.5. Mantenibilidad | | | **X** |  |  | | **Aprobado por el SonarQube** |
| 3.6.5.1. Capacidad de ser analizado | | | **X** |  |  | | **Si por un software externo** |
| 3.6.5.2. Capacidad de ser cambiado | | | **X** |  |  | | **Si** |
| 3.6.5.3. Estabilidad | | | **X** |  |  | |  |
| 3.6.5.4. Capacidad de ser probado | | | **X** |  |  | |  |
| 3.6.5.5 Cumplimiento de la mantenibilidad | | | **X** |  |  | |  |
| 3.6.6. Portabilidad | | |  | **X** |  | | **Posible de instalar en múltiples sistemas operativos pero no se encuentra paquetizado** |
| 3.6.6.1. Adaptabilidad | | | **X** |  |  | |  |
| 3.6.6.2. Instalabilidad | | |  | **X** |  | |  |
| 3.6.6.3. Coexistencia | | | **X** |  |  | |  |
| 3.6.6.4. Capacidad para reemplazar | | | **X** |  |  | |  |
| 3.6.6.5. Cumplimiento de la portabilidad | | | **X** |  |  | |  |
| 3.6.7. Calidad de uso | | |  |  | **X** | | **No aplica** |
| 3.6.7.1. Efectividad | | |  |  | **X** | | **No se ha probado en ambiente de producción real** |
| 3.6.7.2. Productividad | | |  |  | **X** | | **No se ha probado en ambiente de producción real** |
| 3.6.7.3. Seguridad Física | | |  |  | **X** | |  |
| 3.6.7.4 Satisfacción | | |  |  | **X** | | **No se ha probado en ambiente de producción real con usuarios reales** |
| 3.7. Otros requisitos | | |  | **X** |  | | **No pasó la prueba de Seguridad del SonarQube** |
|  | | | | | | | |
| **4. Apéndices** | | |  |  |  | |  |
| Observaciones Generales:  Tanto como el Frontend y el Backend del Sistema de Biblioteca han pasado la prueba del SonarQube con una A en promedio, pero se necesita terminar de validar las funcionalidades reales del proyecto con las historias de usuario y un documento de análisis no proporcionado por el grupo. Hay ciertas correcciones de código que se recomiendan hacer ya que hay un excesivo Código Oloroso y hay puntos del CheckList que no se han podido evaluar debido a que no ha sido probado en un ambiente de producción o no esta mapeado su inspección por el tipo de proyecto. | | | | | | | |
| FIRMA DEL REVISOR :  ***Walter Renzo Muñoz Arellano-15200089*** | | | | | | | |

**LISTAS DE CHEQUEO DE ESTÁNDARES DE FUNCIONES Y PROCEDIMIENTOS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Revisado por** | **Cesar Gutierrez Tineo** | **Cargo o rol** | | | | **Tester** | |
| **Fecha** | **23/10/2019** | **Sitio** | | | | **FISI** | |
| **Nombre del proyecto** | | **Sistema de Biblioteca - FISI** | | | | | |
| **Nombre de la función o procedimiento:** | |  | | | | | |
| **Descripción:**  **El sistema de préstamo de libros está dividido en dos software uno como servicio REST (back-end) y además de otro que es la parte de la interfaz de usuario. Usando Typescript sobre Angular y Javascript sobre NestJS.** | | | | | | | |
| **REVISIÓN DE FUNCIONES Y PROCEDIMIENTOS** | | | | | | | |
| **ACTIVIDAD O ELEMENTO A REVISAR** | | | **SI** | **NO** | **NO APLICA** | | **OBSERVACIONES** |
| 1. ¿El nombre cumple con los estándares?  El nombre debe ser mnemotécnico, no poseer más de 8 caracteres, comenzar por letra, ¿otros? | | |  | x |  | | No usa nomenclatura a pesar de usar un lenguaje altamente tipado. |
| 2. ¿El código cumple con los estándares?  El código debe cumplir con los estándares de programación como modularidad, interrelación entre módulos, secuencia lógica, etc. | | |  | x |  | | Está modularizado pero está muy acoplado. |
| 3. ¿Está la función/ procedimiento documentada?  Cada función, procedimiento, objeto o en general módulo debe especificar que hace y cuál es su función dentro de la aplicación. | | |  | x |  | | No cuenta con documentación de las funciones para identificar el tipo de datos en las funciones. |
| 4. ¿Se ha realizado el proceso de afinamiento de las búsquedas?  Las búsquedas deben ser las necesarias, deben cumplir con su objetivo: deben encontrar lo buscado y deben generar la información requerida. | | |  | x |  | | No se encuentra el uso de índices o afinamientos en consultas SQL |
| 5. ¿Se usan todas las variables, constantes y parámetros? Todo lo creado debe ser utilizado, lo que sobre debe ser eliminado. | | | x |  |  | |  |
| 6. ¿La asignación de valores a las variables, constantes y parámetros tienen un propósito?  Verificar que los procesos sean rápidos, correctos, funcionales y con un alto rendimiento. | | | x |  |  | |  |
| 7. ¿Son correctas las validaciones de condiciones? (ciclos infinitos, división por cero, verificación de rangos, etc.)  Revisar todos los ciclos en el límite, dentro del límite y fuera de este, verificar que no existan divisiones por cero y que los valores tengan una magnitud aceptada por el tipo de dato definido. | | | x |  |  | |  |
| 8. ¿Faltan validaciones?  Verificar cuales validaciones hacen falta: condicionales, estructuras compuestas, procedimientos, funciones… | | | x |  |  | | La validaciones por tipo de valor. |
| 9. ¿Se tienen en cuenta todas las excepciones posibles?  Dentro de cada proceso o procedimiento es necesario preguntarse qué pasaría si ocurriera un caso diferente y verificar si la aplicación está en capacidad de asumir dicho caso sin descontrolarse. | | | x |  |  | |  |
| 10. ¿Las variables que guardan datos de columnas de tablas se han definido de acuerdo con ellas?  Para identificar cada variable y su función, esta debe ser apropiadamente nombrada, para ser fácilmente identificada. | | | x |  |  | |  |
| 11. Si se llaman otras funciones y/o procedimientos, ¿tienen el número de parámetros y el tipo de datos adecuado?  El número y la clase de parámetros deben ser compatibles para no crear errores o conflictos dentro de la aplicación. | | | x |  |  | |  |
| Observaciones Generales:  El código cuenta está bien estructurado y entendible ya que se usa patrones de diseño pero carece de documentación de funciones que el mismo marco de trabajo brinda. | | | | | | | |
| FIRMA DEL REVISOR : Cesar Gutierrez Tineo | | | | | | | |

**LISTAS DE CHEQUEO DE ESTÁNDARES DE PROGRAMACIÓN - CÓDIGO**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Revisado por** | **Juan Alonso Arce Alcántara** | **Cargo o rol** | | | | **Tester** | |
| **Fecha** | **21/10/2019** | **Sitio** | | | | **FISI** | |
| **Nombre del proyecto** | | **Sistema de Biblioteca - FISI** | | | | | |
| **Descripción: Sistema que facilita el préstamo e inventario de los libros que se encuentra en la biblioteca de la Facultad de Ingeniería de Sistemas e Informática.** | | | | | | | |
| **REVISIÓN DE LA PROGRAMACIÓN – CÓDIGO** | | | | | | | |
| **ASPECTO O ELEMENTO A REVISAR** | | | **SI** | **NO** | **NO APLICA** | | **OBSERVACIONES** |
| **1. Código en general** | | | | | | | |
| * 1. ¿Está el código con los espacios establecidos?   Verificar que el código se entienda, que esté bien dividido, que en cada módulo o estructura se pueda reconocer fácilmente en el comienzo y en la terminación. | | | **X** |  |  | | SIGUE LA MISMA ESTRUCTURA DE UN PROYECTO INICIAL |
| * 1. ¿Están ordenadas alfabéticamente las constantes, las variables y los cursores?   En la documentación es importante que las constantes y las variables estén ordenadas alfabéticamente. | | |  | **X** |  | | SE CREARON CONFORME SE NECESITABAN |
| * 1. ¿Están alineados a la izquierda la definición del tipo de dato de las constantes y de las variables?   Este es un estándar de programación que debe cumplirse. | | | **X** |  |  | |  |
| **2. Documentación** | | | | | | | |
| 2.1 ¿Está toda la documentación en una línea diferente al código que se está documentando?  Verificar este aspecto en todo el código. | | |  | **X** |  | | NO TIENE UNA DOCUMENTACIÓN |
| 2.2 ¿Comprende la documentación de funciones/procedimientos tres partes: una descripción general de los que hace la función o procedimiento (módulo), la descripción de los parámetros de entrada y la descripción de los posibles valores y/o parámetros de salida? | | |  | **X** |  | | NO TIENE UNA DOCUMENTACIÓN |
| **3. Parámetros** | | | | | | | |
| 3.1 ¿El nombre de los parámetros es significativo?  Debe ser representativo y fácilmente identificable | | | **X** |  |  | |  |
| **4. Constantes** | | | | | | | |
| 4.1 ¿El nombre de las constantes es significativo?  Debe ser representativo y cumplir con los estándares de programación. | | | **X** |  |  | |  |
| **5. Variables** | | | | | | | |
| 5.1 ¿El nombre de las variables es significativo?  Debe ser representativo y cumplir con los estándares de programación. | | | **X** |  |  | |  |
| 5.2 ¿Están los nombres de las constantes y de las variables alineados a la izquierda junto a la definición del tipo? | | | **X** |  |  | |  |
| Observaciones Generales:  El backend y el frontend siguen la misma estructura inicial que se crea en cualquier proyecto de AngularJS y NestJS, lo cual es bueno para alguien que no tiene mucha experiencia con la tecnología, pueda entender el código y su estructura. | | | | | | | |
| FIRMA DEL REVISOR:  ***Juan alonso Arce Alcántara - 15200232*** | | | | | | | |

**LISTA DE CHEQUEO DE INSPECCIÓN DE CÓDIGO PARA EL LENGUAJE DE PROGRAMACIÓN**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Revisado por** | Bryan Lázaro Cusihuaman | **Cargo o rol** | | | | Tester | |
| **Fecha** | **21/10/19** | **Sitio** | | | | **FISI** | |
| **Nombre del proyecto** | | **Sistema de Biblioteca-Fisi** | | | | | |
| **Nombre de la función o procedimiento** | | **Consultar Obligación** | | | | | |
| **Descripción:**  **Revisión de Código para el lenguaje de programación** | | | | | | | |
| **REVISIÓN DE IMPLEMENTACIÓN DEL LENGUAJE DE PROGRAMACIÓN** | | | | | | | |
| **ASPECTO O ELEMENTO A REVISAR** | | | **SI** | **NO** | **NO APLICA** | | **OBSERVACIONES** |
| 1. ¿El código está escrito siguiendo el estándar de codificación definido para el lenguaje usado? | | | X |  |  | | ESTA BIEN ESTRUCTURADO |
| 2. El código implementa completa y correctamente el diseño detallado correspondiente | | | | | | | |
| 2.1 ¿Hay funciones/procedimientos faltantes? | | |  | X |  | |  |
| * 1. ¿El código de cada función/procedimiento es consistente con su especificación de diseño detallado? | | | X |  |  | |  |
| 3. ¿Se tiene en cuenta condiciones de frontera (valores nulos o negativos)? | | | X |  |  | |  |
| 4. ¿Se tienen en cuenta todas las posibles condiciones de error? | | | X |  |  | |  |
| 5. Los procedimientos para cada invocación a función/procedimiento son correctos | | | | | | | |
| 5.1 ¿Los parámetros de entrada contienen los valores correctos? | | | X |  |  | |  |
| 5.2 ¿Los parámetros de salida tienen el tamaño y tipo correcto? | | | X |  |  | |  |
| 6. ¿Cada sentencia de selección es válida (¿la cláusula “else” está asociada a la sentencia “if*”* correcta?) | | | X |  |  | |  |
| 7. Para cada sentencia de selección múltiple (“switch” , “case of” o una sentencia con el mismo propósito) | | | | | | | |
| 7.1 ¿Los casos considerados están completos? | | |  |  | X | |  |
| 7.2 ¿El caso “default” se considera y se utiliza correctamente? | | |  |  | X | |  |
| 8. Para cada ciclo (while, for, repeat o similares) | | | | | | | |
| 8.1 ¿El ciclo ejecuta el número correcto de iteraciones? | | | X |  |  | |  |
| 8.2 ¿El ciclo no se queda en “loop” infinito? | | | X |  |  | |  |
| 8.3 ¿La lógica de la última iteración del ciclo está completa y correcta? | | | X |  |  | |  |
| 8.4 ¿Hay algún error en la condición que haga que se ejecute una vez más o una vez menos de lo debido? | | |  | X |  | |  |
| 9. Para cada expresión matemática o booleana: | | | | | | | |
| * 1. ¿La precedencia de operadores y de paréntesis son utilizados correctamente? | | | X |  |  | |  |
| * 1. ¿Las conversiones no causan pérdida de información? | | | X |  |  | |  |
| 10. Todos los arreglos y cadenas: | | | | | | | |
| 10.1 ¿Tienen el tamaño correcto? | | | X |  |  | |  |
| 10.2 ¿Están inicializados correctamente? | | | X |  |  | |  |
| 11. Todos los apuntadores usados: | | | | | | | |
| 11.1 ¿Son inicializados antes de hacer referencia a lo apuntado por ellos? | | |  | X |  | |  |
| 11.2 ¿Apuntan a memoria ya liberada? | | |  |  | X | |  |
| 12. ¿Todos los archivos abiertos son cerrados? | | | X |  |  | |  |
| 13. ¿Todos los objetos creados en memoria dinámica son liberados? | | |  |  | X | |  |
| Observaciones Generales:  Agregar algunos comentarios a algunas funciones para una posible ubicación cuando se maneje cambios | | | | | | | |
| FIRMA DEL REVISOR: BRYAN LAZARO CUSIHUAMÁN -14200050 | | | | | | | |