|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

**Proyecto *“Sistema de Conversión y Organización de Documentos Técnicos en Markdown”***

**Documento de Estándares de Programación Versión 1.0**

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |
| Documento de Estándares de Programación |  |

**Historia de Revisión**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Historial de revisiones** |  |  |  |  |
| **Ítem** | **Fecha** | **Versión** | **Descripción** | **Equipo** |
| 1 | 01/12/2024 | 1.0 | Versión Final. |  |

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

**Tabla de Contenidos**

1.OBJETIVO 4

2.DECLARACIÓN DE VARIABLES 5 2.1Descripción de la Variable. 5 2.2Variables de Tipo Arreglo 5

3.Definición de Controles 6 3.1Tipo de datos 6 3.2Prefijo para el Control [6](#_page8_x72.00_y640.01) 3.3Nombre descriptivo del Control 6 3.4Declaración de variables, atributos y objetos 6 3.5Declaración de clases 7 3.6Declaración de métodos [7](#_page11_x72.00_y407.35) 3.7Declaración de funciones [8](#_page13_x72.00_y215.36) 3.8Control de versiones de código fuente 8 3.9Controles ADO.NET [8](#_page15_x72.00_y566.63)

4.Clases. 10 5.Métodos, Procedimientos y Funciones definidos por el Usuario. 10 6.Beneficios 10 7.Conclusiones 11

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |
| Documento de Estándares de Programación |  |

**Estándares de Programación**

1. **OBJETIVOS**

* Garantizar la calidad y consistencia del código: Establecer normas para escribir código limpio y legible, asegurando que todos los desarrolladores sigan una estructura uniforme. Esto facilitará el mantenimiento del sistema web de gestión de pedidos, que integra funcionalidades para la generación de imágenes mediante IA en la imprenta América.
* Asegurar la seguridad en los pedidos y datos de clientes: Definir estándares de seguridad para prevenir vulnerabilidades, especialmente en áreas críticas como el manejo de pagos, los datos personales de los clientes y la protección contra ataques como inyecciones SQL en la base de datos.
* Promover el uso de contenedores y despliegue automatizado: Establecer prácticas para la automatización del despliegue mediante herramientas como GitHub Actions
  + Azure DevOps. Esto asegurará que las actualizaciones en la plataforma de gestión de pedidos y generación de imágenes se implementen rápidamente, sin errores y de manera confiable.
* Facilitar la colaboración en equipo: Fomentar la colaboración entre los desarrolladores adoptando convenciones comunes para la gestión de bases de datos relacionales (como MySQL o PostgreSQL) y NoSQL (como MongoDB). Estas convenciones facilitarán la integración de datos, incluyendo las imágenes generadas y la información de pedidos

1. **DECLARACIÓN DE VARIABLES**

En el sistema web de gestión de pedidos para la imprenta América, las variables deben declararse de manera que se ajusten al propósito requerido, siguiendo criterios que garanticen claridad y consistencia.

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

* Longitud de las Variables:

La longitud de las variables debe ser suficientemente corta para facilitar su manejo, pero lo bastante descriptiva para dejar claro su propósito. Se establece un límite de 16 caracteres para nombres de variables, asegurando legibilidad sin extenderlos innecesariamente.

* Mnemotécnico en los Nombres:

El uso de nombres mnemotécnicos es esencial para hacer que el código sea más fácil de entender. Cada nombre debe reflejar directamente su función en el sistema. Por ejemplo:

* orderID: Identificador único del pedido.
* generatedImage: Imagen creada por la IA.
* deliveryDate: Fecha de entrega del pedido.
* Consistencia:

Mantener un patrón uniforme en toda la aplicación es clave para facilitar el trabajo colaborativo y evitar confusión. Se recomienda usar camelCase para las variables, asegurando coherencia entre los diferentes módulos del sistema.

* Alcance de la variable

Reconocer el alcance de las variables es esencial en un proyecto de gran tamaño. Se puede lograr añadiendo un prefijo que indique el alcance (por ejemplo, g\_ para variables globales y l\_ para variables locales), manteniendo una longitud de nombres adecuada.

Por ejemplo:

* g\_orderList: Lista global de pedidos.
* l\_tempImage: Imagen temporal generada en un proceso local.

|  |  |  |  |
| --- | --- | --- | --- |
| **Alcance** |  | **Prefijo** | **Ejemplo** |
| Global |  | G | gstrNombreUsuario |
| Nivel de la clase |  | *M* | mblnProgresoDelCálculo |
|  |  |  |  |
| Local | del |  | dblVelocidad |
| procedimiento | / | Ninguno |  |
| método |  |  |  |
| Público |  | P | pCantidadUsuario |
| Privado |  | Pr | prCantidadVenta |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |  |  |
| :- | - |  |  |
| Documento de Estándares de Programación |  |  |  |

* El tipo de dato al que pertenece la variable. Por lo tanto la estructura de la variable es como sigue:

|  |  |
| --- | --- |
| **Estructura** | **Descripción de la Variable** |
| LONGITUD. MAX. | 1 16 |
| FORMATO | *Minúscula la primera parte y luego la segunda con Mayúsculas* |
| EJEMPLO | numCuenta |

Siendo el nombre que identifica a la variable: **numCuenta**

1. **Descripción de las Variables utilizadas en el Proyecto.**

En el Sistema de conversión y organización de documentos en markdown, las variables se nombran de manera descriptiva y estarán asociadas a su propósito específico. Esto garantiza claridad y consistencia en el código.

Criterios para el Nombre de las Variables:

* Asociación directa con su funcionalidad: El nombre debe indicar claramente el propósito de la variable dentro del sistema
* Convención de nombres consistente: Se usará el formato camelCase para facilitar la lectura y la colaboración entre los desarrolladores.
* Longitud adecuada: Los nombres serán lo suficientemente cortos para ser manejables, pero descriptivos para evitar ambigüedades.

1. **Variables**

**Variables del documento :**

1. Variables de Identificación y Metadatos:

* document\_id (int) - Identificador único del documento en la base de datos

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

1. title (str) - Título del documento proporcionado por el usuario
2. version (int) - Número de versión del documento (incremental)
3. original\_format (str) - Formato original del archivo subido (ej: "docx", "pdf")
4. created\_at (datetime) - Fecha y hora de creación del documento
5. Variables de Contenido:
   1. markdown\_content (str) - Contenido del documento convertido a formato Markdown
   2. filepath (str) - Ruta temporal del archivo guardado en el servidor
   3. filename (str) - Nombre original del archivo subido
6. Variables de Relaciones:
   1. user\_id (int) - ID del usuario propietario del documento
7. Variables para Mejora de Documentos:
   1. image\_pattern (regex) - Patrón para identificar imágenes en Markdown
   2. extracted\_images (dict) - Diccionario para almacenar imágenes extraídas durante el proceso de mejora
   3. improved\_content (str) - Contenido Markdown después del procesamiento por IA
   4. placeholder\_pattern (regex) - Patrón para marcadores de imágenes temporales
8. Variables de Operaciones:
   1. temp\_dir (str) - Directorio temporal para almacenar archivos durante descargas
   2. skip (int) - Offset para paginación en listados
   3. limit (int) - Límite de resultados por página\*\*

**Variables del Usuario:**

1.Variables de Autenticación y Seguridad:

1. password\_hash (str) - Contraseña del usuario hasheada (usando bcrypt)
2. access\_token (str) - Token JWT generado para sesiones autenticadas
3. secret\_key (str) - Clave secreta para firmar tokens JWT (desde Config)
4. token\_expires (timedelta) - Tiempo de expiración del token (ej: 30 minutos)
5. Variables de Identificación:
   1. user\_id (int) - Identificador único del usuario
   2. username (str) - Nombre de usuario único para login
   3. email (str) - Correo electrónico del usuario

3. Variables de Operaciones:

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |
| Documento de Estándares de Programación |  |

1. plain\_password (str) - Contraseña en texto plano antes de hashing
2. hashed\_password (str) - Resultado del hashing de contraseña
3. expire (datetime) - Fecha/hora de expiración del token
4. existing\_user (User) - Objeto para verificar duplicados en registro
5. Variables de Contexto:
   1. pwd\_context (CryptContext) - Contexto para manejo de hashing (bcrypt)
   2. oauth2\_scheme - Esquema OAuth2 para autenticación Bearer Token

**3. Definición de Controles**

Para nombrar un control en aplicaciones visuales, se debe identificar primero el tipo de control y luego asociarlo con la función que cumple en la aplicación. Esto asegura una nomenclatura coherente y facilita la lectura y mantenimiento del código.

Pasos para Determinar el Nombre de un Control:

* Identificar el tipo de control: Determinar si es un botón, cuadro de texto, lista desplegable, etiqueta, etc.
* Relacionar el control con su función: El nombre debe describir claramente su propósito dentro de la aplicación.
* Utilizar una convención estándar: Se recomienda usar un prefijo que identifique el tipo de control, seguido de un nombre descriptivo que indique su función. Por ejemplo, para un botón relacionado con guardar un pedido: btnGuardarPedido.

1. **Tipo de datos**

|  |  |  |
| --- | --- | --- |
| **Tipo de variable** | **Mnemónico** | **Descripción** |
| <a name="\_page8\_x72.00\_y640.01"></a>Integer | int | Entero de 32 bits con signo. |
| String | st | Cadena de caracteres |
| Float | fl | Coma flotantes, 11-12 dígitos significativos. |
| Double | db | Coma flotante, 64 bits (15-16 dígitos significativos) |
| Object | ob | Objeto genérico |

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

1. **Prefijo para el Control**

El prefijo del control será determinado mediante tres caracteres que estarán conformados por las consonantes más representativas del control, es así, por ejemplo; el control Button, estará asociado al prefijo btn.

|  |  |
| --- | --- |
| Tipo de Control | Prefijo |
| Label | lbl |
| TextBox | txt |
| Button | btn |
| jtable | TABLA[NOMBRE] |

1. **Nombre descriptivo del Control**

Formado por la descripción de la función que lleva a cabo el control, esta debe ser descrita en forma específica y clara.

|  |  |  |
| --- | --- | --- |
| **Tipo de control** | **Prefijo** | **Descripción** |
| TextBox | txt | txtCodbuscar |
| TextBox | txt | txtCodigo |
| TextBox | txt | txtDescripcion |
| TextBox | txt | txtapellido |
| TextBox | txt | txtdni |
| TextBox | txt | txtnombre |
| TextBox | txt | txttelefono |
| TextBox | txt | txtdni |
| TextBox | txt | txtnombre |
| TextBox | txt | txttelefono |
| TextBox | txt | txtdnicli |
| TextBox | txt | txtnrohab |
| TextBox | txt | txtprecio |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |  |
| :- | - |  |
| Documento de Estándares de Programación |  |  |

|  |  |  |
| --- | --- | --- |
| TextBox | txt | txtClave |
| TextBox | txt | txtUsuario |
| TextBox | txt | txtclave |
| TextBox | txt | txtdni |
| TextBox | txt | txtemail |
| TextBox | txt | txtnombre |
| TextBox | txt | txttelefono |
| Button | btn | btnCancelar |
| Button | btn | btnGuardar |

1. **Declaración de variables, atributos y objetos**

1. Se debe declarar una variable por línea.

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | [TipoVariable] [Nombre de la Variable] |
| **Descripción** | <p>Todas las variables o atributos tendrán una longitud máxima de 30 caracteres. </p><p>El nombre de la variable puede incluir más de un sustantivo los cuales se escribirán juntos. </p><p>Si se tuvieran variables que puedan tomar nombres iguales, se le agregará un número asociado (si está dentro de un mismo método será correlativo). </p> |
| **Observaciones** | <p>En la declaración de variables o atributos no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Ejemplo** | <p>Public String nombre </p><p>Indica una variable o atributo que guardará un nombre. </p> |

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

1. **Declaración de clases**

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | [Tipo] Class [Nombre de Clase] |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. Tipo se refiere a si la clase será: Private, Public o Protected. |
| **Observaciones** | <p>En la declaración de clases no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Ejemplo** | Private Class Usuario Indica una clase Usuario |

|  |  |
| --- | --- |
| <a name="\_page11\_x72.00\_y407.35"></a>**Título** | **Descripción** |
| **Sintaxis** | internal class Conexion |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. |
| **Observaciones** | <p>En la declaración de clases no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Ejemplo** | En esta clase se realizará la conexión a la base de datos SQL. |

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |
| Documento de Estándares de Programación |  |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | <p>- **clsECargo** </p><p>- **clsECliente** </p><p>- **clsEHabitaciones** </p><p>- **clsEReservas** </p><p>- **clsEUsuarios** </p> |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. |
| **Observaciones** | <p>En la declaración de clases no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Descripción General** | Todas las clases estarán contenidas en la Capa ‘’.Datos’’ |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | **RenderTabla clsNCargo clsNCliente clsNHabitaciones clsNReservas clsNUsuarios** |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. |
| **Observaciones** | <p>En la declaración de clases no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. </p> |

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

|  |  |
| --- | --- |
|  | ● Caracteres tildados: á, é, í, ó, ú. |
| **Descripción General** | Todas las clases estarán contenidas en la Capa ‘’.Entidad’’,las cuales tendrán los tipos de variables que se usaran junto a la base de datos. |

1. **Declaración de metodos**
2. **Declaración<a name="\_page13\_x72.00\_y215.36"></a> de funciones**

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | <p>- **public void inicio()** </p><p>- **public void Limpiar()** </p><p>- **private void MostrarTabla()** </p> |
| **Descripción** | <p>El nombre del objeto consta hasta de 25 caracteres, no es necesario colocar un nombre que indique la clase a la cual pertenece. </p><p>La primera letra de la primera palabra del nombre será escrita en mayúsculas </p><p>El tipo de dato de retorno se coloca al final y será obligatorio colocarlo. </p> |
| **Observaciones** | <p>En la declaración de objetos no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Descripción General** | Todas las funciones estarán contenidas en dentro del formulario “FrmUsuarios”, los cuales al ser ejecutadas permitirán su funcionamiento. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | - private void MostrarTabla() |
| **Descripción** | <p>El nombre del objeto constará hasta de 25 caracteres, no es necesario colocar un nombre que indique la clase a la cual pertenece. </p><p>La primera letra de la primera palabra del nombre será escrita en mayúsculas </p> |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |
| :- | - |
| Documento de Estándares de Programación |  |

|  |  |
| --- | --- |
|  | El tipo de dato de retorno se coloca al final y será obligatorio colocarlo. |
| **Observaciones** | <p>En la declaración de objetos no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Descripción General** | Todas las funciones estarán contenidas en dentro del formulario “FrmPrincipal”, los cuales al ser ejecutadas permitirán su funcionamiento. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | - private void MtdLimp() |
| **Descripción** | <p>El nombre del objeto consta hasta de 25 caracteres, no es necesario colocar un nombre que indique la clase a la cual pertenece. </p><p>La primera letra de la primera palabra del nombre será escrita en mayúsculas </p><p>El tipo de dato de retorno se coloca al final y será obligatorio colocarlo. </p> |
| **Observaciones** | <p>En la declaración de objetos no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Descripción General** | Todas las funciones estarán contenidas en dentro del formulario “FrmListaHab”, los cuales al ser ejecutadas permitirán su funcionamiento. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | <p>- public void bloqueo() </p><p>- public void accion(int n,JButton a) </p><p>- private void actualizo\_estado() </p> |
| **Descripción** | El nombre del objeto consta hasta de 25 caracteres, no |

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

|  |  |
| --- | --- |
|  | <p>es necesario colocar un nombre que indique la clase a la cual pertenece. </p><p>La primera letra de la primera palabra del nombre será escrita en mayúsculas </p><p>El tipo de dato de retorno se coloca al final y será obligatorio colocarlo. </p> |
| **Observaciones** | <p>En la declaración de objetos no se deberá utilizar caracteres como: </p><p>- Letra Ñ o ñ. </p><p>- Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. </p><p>- Caracteres tildados: á, é, í, ó, ú. </p> |
| **Descripción General** | Todas las funciones estarán contenidas en dentro del formulario “FrmHabitaciones”, los cuales al ser ejecutadas permitirán su funcionamiento. |

1. **Control de versiones de código fuente**

Cada modificación realizada será guardada de la forma:

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Formato** | [Reserva][ \_ ][FECHA][ \_ ][HORA] donde y la fecha estará en formato yyyymmdd y la hora en formato HHMM. |
| **Descripción** | Se generarán archivos con las siguientes extensiones: PDF O EXCEL.Por ejemplo: Reserva |

1. **Controles<a name="\_page15\_x72.00\_y566.63"></a> ADO.NET**

Objetos de ADO.NET Aunque hay miles de objetos disponibles como parte de .NET, es probable que se use ADO.NET como parte de las aplicaciones, por lo tanto algunos estándares para nombrar los objetos de ADO.NET más comunes. A continuación, se listan los prefijos que se utiliza:

|  |  |
| --- | --- |
| **Componente** | **Prefijo** |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |
| :- | - |
| Documento de Estándares de Programación |  |

|  |  |
| --- | --- |
| **DataSet** | Ds |
| **DataTable** | Dt |
| **DataView** | Dv |
| **DataRow** | Drw |
| **Connection\*** | Cnn |
| **Command\*** | Cmd |
| **DataAdapter\*** | Da |
| **CommandBuilder\*** | Bld |
| **DataReader\*** | Dr |

Ejemplos: de declaración de los objetos ADO.net

* drEmps As New SqlDataReader()
* drCust As New SqlDataReader()
* dsEmps As DataSet
* dsCust As DataSet
* System.Data.SqlClient
* System.Data
* System.Threading.Tasks

1. **Clases.**

El nombre de las clases debe ser autodescriptivo de manera que no se requiera, en lo posible, entrar al código de la función para saber qué es lo que realiza.

El estándar para nombres de clases es usar iniciar con las siglas **cls,** la cual debe estar escrita en minúscula seguido del nombre que identifica la clase, la primera letra del nombre debe iniciar con mayúscula.

Nombre de las Clases utilizadas en el Proyecto:

* **clsDocument**
* **clsUser**

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

**Nota:**

* No se hará uso de los caracteres: Espacio en blanco " ", Caracter de subrayado "\_".

1. **Métodos, Procedimientos y Funciones definidos por el Usuario.**

El nombre de las funciones y procedimientos debe ser autodescriptivo de manera que no se requiera, en lo posible, entrar al código de la función para saber qué es lo que realiza. ***verbo-Sustantivo***

El estándar para nombres de procedimiento es usar un Verbo que describa la acción realizada seguida por un sustantivo (objeto sobre el cual actúa). Se recomienda:

* Usar un nombre que represente una acción y un objeto. El nombre del procedimiento debe indicar qué hace el procedimiento a... o qué hace el procedimiento con....
* El verbo debe estar en infinitivo.
* Ser consistente en el orden de las palabras. Si se va a usar ***verboNombre***, siempre usar ***verboNombre***.
* Ser consistente en los verbos y sustantivos usados. Por ejemplo, si tiene un procedimiento ***asignarNombre***, en vez de ***colocarNombre***.
* Para la acción **modificar cuenta del usuario** se define: **modificar Cuenta**
* Verbo: modificar
* Sustantivo: Cuenta
* **Nota:**
* No se hará uso de los caracteres: Espacio en blanco " ", Caracter de subrayado "\_".
* La nomenclatura de argumentos o parámetros pasados a los procedimientos/funciones así como para valores devueltos por funciones sigue las mismas convenciones que la nomenclatura para variables.

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión 1.0 |
| Documento de Estándares de Programación |  |

1. **Beneficios**

* La documentación hará que el código del sistema de gestión de pedidos para la Imprenta América sea más fácil de entender. Esto permitirá que cualquier miembro del equipo comprenda rápidamente cómo interactúan los distintos componentes de la aplicación, incluyendo la integración de la IA para la generación de imágenes. Este enfoque también facilita el mantenimiento y la actualización del sistema.
* Una adecuada documentación y estructuración del código permitirá la reutilización de módulos y funciones. Por ejemplo, la lógica para gestionar pedidos o para generar imágenes mediante IA puede ser reutilizada tanto en este proyecto como en futuros desarrollos, eliminando la necesidad de escribir código desde cero y ahorrando tiempo de desarrollo.
* Documentar el código ayuda a detectar partes que podrían necesitar ser reescritas. Si una sección del código es difícil de explicar en los comentarios, probablemente esté mal estructurada o sea innecesariamente compleja. Esto facilitará su reestructuración, mejorando la claridad, eficiencia y desempeño de la aplicación
* El proyecto define una estructura clara para las variables que almacenan información conectada a la base de datos SQL. Estas variables se utilizarán para gestionar información clave, como los pedidos de impresión, los datos de los clientes y los archivos generados por la IA. Esto asegura un manejo seguro y eficiente de la información en todo el sistema.

|  |  |
| --- | --- |
| SISTEMA DE CONVERSIÓN Y ORGANIZACIÓN DE DOCUMENTOS EN MARKDOWN | Versión: 1.0 |
| Documento de Estándares de Programación |  |

1. **Conclusiones**

* Las bases de datos relacionales son fundamentales para gestionar información transaccional como los detalles de los pedidos, datos de los clientes y estados de las operaciones, garantizando la integridad y seguridad de la información
* Las bases de datos no relacionales ofrecen una solución eficiente para manejar datos como imágenes generadas por la IA, preferencias de diseño de los clientes o registros históricos de pedidos, asegurando escalabilidad en el sistema.
* La utilización de herramientas de automatización simplifica el despliegue continuo de actualizaciones, reduciendo errores y optimizando la eficiencia operativa en el manejo del sistema.  
  Curso Programación III Página 21 de 21