Estándar de Programación

Lenguaje C#

**Proyecto *“Sistema Web para mejorar el aprendizaje de los estudiantes del VI ciclo de educación secundaria en el curso de Educación para el Trabajo”***

Curso: *Programación Web II*

Docente: *Mag. Patrick Cuadros Quiroga*

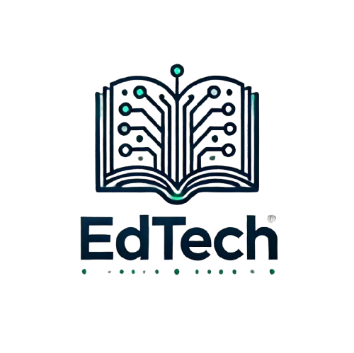
Integrantes:

***Concha Llaca, Gerardo Alejandro*** ***(2017057849)***

***Ticahuanca Cutipa, Fiorela Milady*** ***(2020068765)***

***Cuadros Napa, Raúl Marcelo*** ***(2017057851)***







*El proyecto “Sistema Web para mejorar el aprendizaje de los estudiantes del VI ciclo de educación secundaria en el curso de Educación para el Trabajo” tiene como objetivo principal proporcionar una plataforma interactiva que facilite el acceso a recursos educativos, optimizando el proceso de enseñanza-aprendizaje. Este sistema está diseñado para ofrecer a los estudiantes herramientas digitales que potencien sus conocimientos y habilidades prácticas en el área de Educación para el Trabajo.*

*La implementación de este sistema web permite a los docentes gestionar materiales de aprendizaje, realizar evaluaciones en línea y dar seguimiento al progreso de los estudiantes. Por otro lado, los estudiantes podrán acceder a contenidos actualizados, realizar prácticas interactivas y recibir retroalimentación en tiempo real. El sistema se desarrolla utilizando tecnologías modernas como Java, garantizando así un software robusto, escalable y de fácil mantenimiento.*

*Este documento refleja los estándares de codificación del lenguaje, no solo busca definir la nomenclatura de las variables, objetos, métodos y funciones, sino que también hace referencia al orden y legibilidad del código escrito.*

Estándar de Programación Lenguaje C#

# Nombres de los archivos

Los archivos de código fuente en C# deben seguir una convención clara y coherente para facilitar su identificación y mantenimiento.

# Extensiones de Archivos

|  |  |
| --- | --- |
| **Tipo de Archivo** | **Extensión** |
| Código fuente en C# | .cs |
| Archivos de proyecto | .csproj |
| Configuración de aplicación | .json, .xml |

# Organización de los archivos

Cada archivo debe contener una única clase pública o interfaz pública. Si existen clases auxiliares o privadas relacionadas, pueden incluirse en el mismo archivo, siempre que no excedan las 2000 líneas de código.

/SistemaWebEducativo

│ ├── /Controllers

│ ├── /Models

│ ├── /Services

│ ├── /Data

│ ├── /Pages

│ └── /wwwroot

└── appsettings.json

# Descripción de Carpetas

* **Controllers:** Manejan la lógica de interacción entre el usuario y el sistema.
* **Models:** Contiene las entidades que representan las tablas de la base de datos.
* **Services:** Lógica de negocio y acceso a los datos.
* **Data:** Contexto de la base de datos (Entity Framework).
* **Pages:** Páginas Razor para la interfaz de usuario.
* **wwwroot:** Archivos estáticos (CSS, JS, imágenes).

# Estructura de un Archivo C#

El orden recomendado dentro de un archivo C# es:

1. Comentarios de inicio.
2. Directivas using.
3. Declaración del espacio de nombres (namespace).
4. Declaración de la clase o interfaz.
5. Campos estáticos.
6. Campos de instancia.
7. Constructores.
8. Propiedades.
9. Métodos (orden: públicos, protegidos, privados).

**Ejemplo Base de un Archivo C#**

/\*

\* Usuario.cs

\* Proyecto: Sistema Web Educativo

\* Autor: [Tu Nombre]

\* Fecha: [Fecha de Creación]

\* Descripción: Representa la entidad Usuario

\*/

using System.ComponentModel.DataAnnotations;

namespace Financiera.WebApp.Modelos

{

/// <summary>

/// Representa un usuario del sistema.

/// </summary>

public class Usuario

{

// Campos estáticos

// Campos de instancia

/// <summary>

/// Identificador único del usuario.

/// </summary>

[Key]

public int Id { get; set; }

/// <summary>

/// Nombre del usuario.

/// </summary>

[Required]

[MaxLength(100)]

public string Nombre { get; set; }

/// <summary>

/// Apellido del usuario.

/// </summary>

[Required]

[MaxLength(100)]

public string Apellido { get; set; }

/// <summary>

/// Correo electrónico del usuario.

/// </summary>

[Required]

[EmailAddress]

public string Email { get; set; }

/// <summary>

/// Contraseña del usuario (encriptada).

/// </summary>

[Required]

public string Contraseña { get; set; }

/// <summary>

/// Rol del usuario (Admin, Docente, Estudiante).

/// </summary>

[Required]

public string Rol { get; set; }

/// <summary>

/// Constructor por defecto.

/// </summary>

public Usuario() { }

/// <summary>

/// Constructor con parámetros.

/// </summary>

public Usuario(string nombre, string apellido, string email, string contraseña, string rol)

{

Nombre = nombre;

Apellido = apellido;

Email = email;

Contraseña = contraseña;

Rol = rol;

}

}

}

# Convenciones de Nomenclatura

* **Clases y Métodos:** PascalCase (e.g., Usuario, ObtenerCursos()).
* **Propiedades:** PascalCase (e.g., Nombre, Apellido).
* **Campos Privados:** camelCase con guion bajo (e.g., \_contador).
* **Constantes:** UPPER\_CASE (e.g., MAX\_INTENTOS).

# Validaciones y Anotaciones

Utiliza DataAnnotations para validar los campos:

* [Required] para campos obligatorios.
* [MaxLength(100)] para límites de longitud.
* [EmailAddress] para validar correos electrónicos.

Ejemplo:

[Required(ErrorMessage = "El nombre es obligatorio")]

[MaxLength(100, ErrorMessage = "Máximo 100 caracteres")]

public string Nombre { get; set; }

# Comentarios Especiales

* XXX: Indica que algo tiene un error pero funciona.
* FIXME: Indica que algo tiene un error y no funciona.

# Entidades de la Base de Datos

|  |  |  |
| --- | --- | --- |
| Tabla | Descripción | Nomenclatura |
| Usuarios | Almacena la información de usuarios | USR |
| Cursos | Información de los cursos | CRS |
| EstudiantesCursos | Relación entre estudiantes y cursos | EST\_CRS |
| Materiales | Almacena materiales de los cursos | MAT |
| Evaluaciones | Contiene las evaluaciones | EVA |
| Preguntas | Almacena las preguntas | PRG |
| Respuestas | Almacena las respuestas | RSP |
| Notas | Contiene las calificaciones | NOT |
| Asistencias | Registra la asistencia de alumnos | AST |

# Ejemplo de Modelo con Relaciones

public class Curso

{

[Key]

public int Id { get; set; }

[Required]

public string Nombre { get; set; }

[Required]

public string Descripcion { get; set; }

public int DocenteId { get; set; }

[ForeignKey("DocenteId")]

public Usuario Docente { get; set; }

public List<Material> Materiales { get; set; }

}