Redwin Valverde Castro

InA San Ramón | Nelson jiménez jiménez

Proyecto Fase 2 – POO

Programa: Programador(a) de aplicaciones informáticas Módulo: Programación Orientada a Objetos Código: CSTI0208

# Tabla de contenidos

**Contenido**

[Tabla de contenidos 2](#_Toc92888696)

[Introducción 3](#_Toc92888697)

[Objetivos 4](#_Toc92888698)

[Problema Empresarial 5](#_Toc92888699)

[Diseño de la Aplicación 6](#_Toc92888700)

[Historias de Usuario: 6](#_Toc92888701)

[Historia 1: 6](#_Toc92888702)

[Historia 2: 8](#_Toc92888703)

[Historia 3: 10](#_Toc92888704)

[Historia 4: 12](#_Toc92888705)

[Requerimientos: 14](#_Toc92888706)

[Prioridades: 15](#_Toc92888707)

[Diagrama de clases: 16](#_Toc92888708)

[Diagrama Lógico de la Base de Datos: 17](#_Toc92888709)

[Diccionario de Datos de la Base de Datos: 18](#_Toc92888710)

[Conclusiones 23](#_Toc92888711)

[Cambios para la Fase 2 24](#_Toc92888712)

# Introducción

En este trabajo se continuarán los problemas del módulo del módulo 3 (bases de datos), utilizando la base de datos desarrollada para el proyecto creado para el Instituto Especializado de Entrenamiento de Triatlón (IEET), se usarán los objetivos dados en la hoja de instrucciones junto con los problemas empresariales que se emplearán en las siguientes páginas, las historias de usuario necesarias para cada una de las visiones necesarias junto con sus criterios de aceptación, tareas asignadas y reglas de negocio, requerimientos para describir que debería hacer la aplicación para que sea funcional en la mayoría de casos de uso, un diagrama de clases que describa la estructura y sus atributos, un diagrama lógico de la base de datos que se base en el diagrama de clases anterior, un diccionario de datos de la base de datos ya realizado en el módulo 3 (bases de datos) y por último, conclusiones.

Cada una de lo dicho anteriormente se tomará en cuenta con lo recientemente aprendido en el módulo 5 (Programación Orientada a Objetos).

# Objetivos

1. Crear diagramas de clases UML utilizando entidades, atributos, métodos, herencia, polimorfismo, sobrecarga para definir claramente la estructura del problema.
2. Utilizar el marco de trabajo Scrum para definir prioridades, requerimientos, tareas asociadas, historias de usuario para establecer las necesidades de usuario.
3. Utilizar Microsoft SQL Server Management Studio para definir la estructura de la base de datos.
4. Crear en Visual Studio el modelo de capas para la solución de la aplicación, diseñando los formularios, capas de lógica, capas de acceso a datos y capas de entidades requeridas, para crear una aplicación de escritorio y una aplicación web.

# Problema Empresarial

El Instituto Especializado de Entrenamiento de Triatlón (IEET) requiere un sistema que administre los procesos de administración de institutos, matrículas de atletas, administración de programas y módulos y administración de entrenadores(as).

Se requiere que el sistema sea capaz de modificar, almacenar y mostrar datos de los principales procesos requeridos para la institución además de tener un ordenamiento y visualización adecuado a las siguientes historias.

Para siempre tener actualizada la información de los institutos, módulos, programas, entrenadores, atletas y matrículas con una interfaz llamativa y adecuada al instituto.

# Diseño de la Aplicación

## Historias de Usuario:

### Dibujo de personaje amarillo Descripción generada automáticamente con confianza bajaHistoria 1:

Como

Encargado(a) del proceso de matrícula de atletas. (Recepcionista)

Quiero

Crear, modificar y deshabilitar atletas.

Para poder

Tener mejor ordenamiento a la hora de crear o modificar un nuevo atleta además deshabilitarlo si es necesario.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número**: 1 | **Usuario**: Encargado(a) del proceso de inscripción de atletas. (Recepcionista) | |
| **Nombre historia**: Ingreso, modificación y deshabilitación de atletas. | | |
| **Prioridad en negocio**:  Alta. | | **Riesgo en desarrollo**:  Medio. |
| **Puntos estimados**: 30 Horas. | | **Iteración asignada**: 3. |
| **Programador responsable**: Redwin Valverde Castro. | | |
| **Descripción**:  Quiero administrar la información de los atletas. | | |
| **Criterios de aceptación**:   * El cliente puede agregar nuevos atletas, modificarlos o deshabilitarlos y además realizar búsquedas dentro de ellos. * Cuando se ingrese un nuevo atleta debe ser obligatorio ingresar la identificación, el nombre, apellidos, la fecha de nacimiento, el número de teléfono 1, el email 1, el distrito, cantón, provincia y país, además de la dirección exacta, en caso contrario, no será necesario ingresar un número de teléfono 2 ni un email 2, además de una observación. * Deberá poderse activar o desactivar la actividad de un atleta, en caso de que deje el instituto y quiera volver luego. * Las búsquedas serán por medio del número de identificación o nombre del atleta. * En el momento de modificación o creación de algún atleta, deberá aparecer una pantalla de confirmación de los datos a modificar o agregar. | | |

**Tareas asignadas a la historia**

1. Crear las tablas necesarias para ingresar muchos atletas junto con sus matrículas.
2. Programar procedimientos CRUD a las tablas de la base de datos.
3. Crear una interfaz de usuario para el mantenimiento de los atletas que funcione en Windows, además de una pantalla de confirmación al crear o modificar sus datos.

**Reglas de negocio:**

1. Como encargado(a) del proceso de inscripción de atletas, quiero que al momento de ingresar un nuevo atleta sea obligatorio ingresar la identificación, el nombre, apellidos, la fecha de nacimiento, el número de teléfono 1, el email 1, el distrito, cantón, provincia y país, además de la dirección exacta, en caso contrario, no será necesario ingresar un número de teléfono 2 ni un email 2, además de una observación.
2. Un atleta puede ser deshabilitado, ya que, puede estar ausente o pasarse a otra institución.
3. Quiero que la interfaz de usuario sea amigable con el usuario además de llevar los colores de la institución.

### Dibujo de personaje amarillo Descripción generada automáticamente con confianza bajaHistoria 2:

Como

Encargado(a) del proceso de matrícula de atletas. (Recepcionista)

Quiero

Crear y modificar matrículas.

Para poder

Tener mejor ordenamiento a la hora de crear o modificar una nueva matrícula.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número**: 2 | **Usuario**: Encargado(a) del proceso de matrícula de atletas. (Recepcionista) | |
| **Nombre historia**: Ingreso y modificación de matrículas. | | |
| **Prioridad en negocio**:  Alta. | | **Riesgo en desarrollo**:  Medio. |
| **Puntos estimados**: 30 Horas. | | **Iteración asignada**: 3. |
| **Programador responsable**: Redwin Valverde Castro. | | |
| **Descripción**:  Quiero administrar la información de las matrículas. | | |
| **Criterios de aceptación**:   * El cliente puede agregar nuevos matriculas, modificarlos o deshabilitarlos y además realizar búsquedas dentro de ellas. * Cuando se ingrese una nueva matrícula debe ser obligatorio ingresar el código del atleta creado anteriormente, la fecha de la matrícula, el estado, el monto cancelado, el tipo de cobro, el tipo de pago y no es obligatorio ingresar un detalle de matrícula. * En caso de que un atleta sea inactivo, su estado de matrícula pasará a ser “abandono”. * Las búsquedas serán por medio del número de identificación del atleta. * En el momento de modificación o creación de alguna matrícula, deberá aparecer una pantalla de confirmación de los datos a modificar o agregar. | | |

**Tareas asignadas a la historia**

1. Crear las tablas necesarias para ingresar matrículas.
2. Programar procedimientos CRUD a las tablas de la base de datos.
3. Crear una interfaz de usuario para el mantenimiento de las matrículas que funcione en Windows, además de una pantalla de confirmación al crear o modificar sus datos.

**Reglas de negocio:**

1. Como encargado(a) del proceso de matrícula, quiero que al momento de ingresar una nueva matrícula sea obligatorio ingresar el código del atleta creado anteriormente, la fecha de la matrícula, el estado, el monto cancelado, el tipo de cobro, el tipo de pago y no es obligatorio ingresar un detalle de matrícula.
2. Una vez que un atleta sea deshabilitado, su estado en matricula pasará a ser “abandono”.
3. Quiero que la interfaz de usuario sea amigable con el usuario además de llevar los colores de la institución.

### Dibujo de personaje amarillo Descripción generada automáticamente con confianza bajaHistoria 3:

Como

Administrador(a) del Instituto.

Quiero

Crear, modificar y deshabilitar programas y módulos.

Para poder

Tener mejor ordenamiento a la hora de crear o modificar un nuevo módulo o programa.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número**: 3 | **Usuario**: Administrador(a) del Instituto. | |
| **Nombre historia**: Ingreso, modificación y deshabilitación de módulos abiertos. | | |
| **Prioridad en negocio**:  Alta. | | **Riesgo en desarrollo**:  Alto. |
| **Puntos estimados**: 50 Horas. | | **Iteración asignada**: 4 |
| **Programador responsable**: Redwin Valverde Castro. | | |
| **Descripción**:  Quiero administrar la información de los módulos abiertos. | | |
| **Criterios de aceptación**:  El cliente puede agregar nuevos módulos abiertos, además de modificarlos.   * Cuando se ingrese un nuevo módulo abierto debe ser obligatorio ingresar el código del entrenador que lo impartirá, el código del módulo, el código del horario (se mostrará en pantalla los horarios), el costo del mismo, la fecha de inicio y la fecha de finalización se calculará automáticamente. * Las búsquedas serán por medio del código del módulo abierto. * En el momento de modificación o creación de algún módulo abierto, deberá aparecer una pantalla de confirmación de los datos a modificar o agregar. | | |

**Tareas asignadas a la historia**

1. Crear las tablas necesarias para ingresar módulos y programas.
2. Programar procedimientos CRUD a las tablas de la base de datos.
3. Crear una interfaz de usuario para el mantenimiento de los módulos y programas que funcione en Windows, además de una pantalla de confirmación al crear o modificar sus datos.

**Reglas de negocio:**

1. Como encargado(a) del proceso de ingreso y modificación de módulos abiertos, quiero que sea obligatorio ingresar el código del entrenador que lo impartirá, el código del módulo, el código del horario (se mostrará en pantalla los horarios), el costo del mismo, la fecha de inicio y la fecha de finalización se calculará automáticamente.
2. Quiero que la interfaz de usuario sea amigable con el usuario además de llevar los colores de la institución.

### Dibujo de personaje amarillo Descripción generada automáticamente con confianza bajaHistoria 4:

Como

Administrador(a) del Instituto.

Quiero

Agregar, modificar y consultar entrenadores.

Para poder

Tener actualizada la información de las personas entrenadoras.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número**: 4 | **Usuario**: Administrador(a) del Instituto. | |
| **Nombre historia**: Ingreso, modificación y deshabilitación de entrenadores. | | |
| **Prioridad en negocio**:  Alta. | | **Riesgo en desarrollo**:  Alto. |
| **Puntos estimados**: 30 Horas. | | **Iteración asignada**: 2 |
| **Programador responsable**: Redwin Valverde Castro. | | |
| **Descripción**:  Quiero administrar la información de los entrenadores. | | |
| **Criterios de aceptación**:   * El cliente puede agregar nuevos entrenadores, modificarlos o deshabilitarlos y además realizar búsquedas dentro de ellos. * Cuando se ingrese un nuevo entrenador debe ser obligatorio ingresar el nombre, los apellidos, el horario al que asistirán (se mostrarán los horarios a un costado para mayor facilidad), el número de identificación, la fecha de nacimiento, el teléfono 1, el email 1, el distrito, cantón, provincia y el país de residencia, además de la dirección exacta y su cuenta bancaria, de lo contrario no será obligatorio ingresar un teléfono 2 y un email 2. * Las búsquedas serán por medio del número de identificación o nombre o ambos. * Se deberá llevar un registro de las horas laboradas de cada entrenador. * En el momento de modificación o creación de algún entrenador, deberá aparecer una pantalla de confirmación de los datos a modificar o agregar. | | |

**Tareas asignadas a la historia**

1. Crear las tablas necesarias para ingresar nuevos entrenadores(as).
2. Programar procedimientos CRUD a las tablas de la base de datos.
3. Crear una interfaz de usuario para el mantenimiento de los entrenadores(as), que funcione en cualquier plataforma, además de una pantalla de confirmación al crear o modificar sus datos.

**Reglas de negocio:**

1. Como encargado(a) del ingreso y modificación entrenadores, quiero que sea obligatorio ingresar el nombre, los apellidos, el horario al que asistirán, el número de identificación, la fecha de nacimiento, el teléfono 1, el email 1, el distrito, cantón, provincia y el país de residencia, además de la dirección exacta y su cuenta bancaria, de lo contrario no será obligatorio ingresar un teléfono 2 y un email 2.
2. Los entrenadores pueden deshabilitarse por motivos de actividad.
3. Quiero que la interfaz de usuario sea amigable con el usuario además de llevar los colores de la institución.

## Requerimientos:

El sistema debe hacer el mantenimiento (insertar, actualizar, deshabilitar, modificar datos) de las siguientes tablas: Institutos, Entrenadores, Vacaciones\_Incapacidades\_Eventos, Horarios\_Entrenadores, Entrenadores, Modulos\_Elegidos, Registro\_Horas\_Laboradas, Modulos, Programas, Modulos\_Abiertos, Horarios\_Modulos, Atletas, Matriculas, Detalle\_Matricula.

El sistema debe realizar el proceso de:

1. Abrir un nuevo instituto, solicitando de forma obligatoria los datos del nombre, el número de teléfono 1, el email 1, sitio web, el horario que tendrá, el distrito, cantón, provincia y país, además de la dirección exacta, en caso contrario, no será necesario ingresar un número de teléfono 2 ni un email 2, realizando las comprobaciones de que se ingrese la provincia correctamente.
2. Agregar un nuevo atleta, solicitando de forma obligatoria la identificación, el nombre, apellidos, la fecha de nacimiento, el número de teléfono 1, el email 1, el distrito, cantón, provincia y país, además de la dirección exacta, en caso contrario, no será necesario ingresar un número de teléfono 2 ni un email 2, además de una observación, realizando las comprobaciones de que se ingrese el número de cédula único y que la provincia sea correcta.
3. En el caso de la matrícula, se solicita de forma obligatoria la fecha, el estado, el monto cancelado, el tipo de cobro y el tipo de pago, como no obligatorio estará el detalle de matrícula, que servirá como un agregado para el usuario. Realizando las comprobaciones de la fecha sea mayor o igual a la fecha actual, de que el estado de la matrícula sea “En Curso”, “Abandono”, “Aprobado” o “Reprobado”, de que el monto cancelado sea mayor a 0, de que el tipo de cobro sea “Curso”, “Curso y Matrícula” y de que el tipo de pago sea “Transferencia”, ”Sinpe”, “Tarjeta”, “Efectivo”, “Otro”.
4. Agregar un nuevo módulo abierto, se solicitará de forma obligatoria el código del entrenador al que será asignado, el código del módulo que se impartirá, el código del horario, el costo de dicho módulo, por último, no será obligatorio ingresar la fecha de inicio, la fecha de finalización se calculará automáticamente.
5. Al agregar un nuevo entrenador se solicitará de forma obligatoria el nombre, los apellidos, el horario al que asistirán, el número de identificación, la fecha de nacimiento, el teléfono 1, el email 1, el distrito, cantón, provincia y el país de residencia, además de la dirección exacta y su cuenta bancaria, de lo contrario no será obligatorio ingresar un teléfono 2 y un email 2. Realizando las comprobaciones de que el número de identificación sea único y de que la provincia sea correcta.
6. Luego se le dará un horario a impartir, cada uno de los horarios ya deben estar creados con anterioridad.

## Prioridades:

1. El sistema será capaz de abrir un nuevo módulo y modificarlo, ingresar un nuevo entrenador(a) y modificarlo(a), además de un nuevo atleta y ser modificado(a),
2. El programa tendrá la capacidad de ingresar los datos del instituto además de poder ser modificados.
3. Se crearán horarios predefinidos para los módulos y los entrenadores(as) para que en el momento de crear alguno de los dos, solamente se seleccione un horario esperado.
4. Los módulos, programas y el instituto serán creados con anterioridad
5. Se podrá ingresar los datos del entrenador ya sean necesarios o no, podrán ser modificados en cualquier momento para dar su respectivo mantenimiento.
6. Ingresar un nuevo atleta con sus respectivos datos conllevará a ingresarlo a una matrícula, ambos con datos a llenar que también pueden ser modificados en cualquier momento.
7. El programa llevará un registro de las vacaciones, incapacidades y eventos de cada uno de los entrenadores(as), además, de un registro de las horas laboradas.
8. El entrenador será ingresado al módulo abierto que impartirá.
9. Crear un programa con los colores de la institución junto con el logo para mostrar que es realmente de la institución.
10. Al ingresar a la aplicación, mostrará un menú con cuadros de cada una de las opciones (institutos, profesores, módulos abiertos, atletas), dentro de ellas, eliges que es lo que quieres hacer (modificar, agregar, deshabilitar).

## Diagrama de clases:

## Diagrama Descripción generada automáticamente con confianza mediaDiagrama Lógico de la Base de Datos:

## Diccionario de Datos de la Base de Datos:

|  |  |  |  |
| --- | --- | --- | --- |
| INSTITUTOS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| ID\_Instituto | tinyint | x | PK, IDENTITY |
| Nombre\_Instituto | varchar(30) | x |  |
| Telefono1\_Instituto | varchar(30) | x |  |
| Telefono2\_Instituto | varchar(30) |  |  |
| Email1\_Instituto | varchar(30) | X |  |
| Email2\_Instituto | varchar(30) |  |  |
| SitioWeb\_Instituto | varchar(150) | X |  |
| Horario\_Instituto | varchar(30) | X |  |
| Distrito\_ Instituto | varchar(30) | X |  |
| Canton\_ Instituto | varchar(30) | X |  |
| Provincia\_ Instituto | varchar(10) | X | Valores: San José, Alajuela, Cartago, Heredia, Guanacaste, Puntarenas, Limón |
| País\_ Instituto | varchar(20) | X |  |
| Direccion\_ Instituto | varchar(150) | x |  |
| Activo\_Instituto | bit | x |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ENTRENADORES | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Entrenador | int | x | PK, IDENTITY |
| Cod\_Horario\_Entr | int | x | FK |
| ID\_Instituto | tinyint | X | FK |
| ID\_Entrenador | varchar(30) | X | UNIQUE |
| Nombre\_ Entrenador | varchar(20) | X |  |
| Apellido1\_ Entrenador | varchar(20) | X |  |
| Apellido2\_ Entrenador | varchar(20) | X |  |
| Nacimiento\_ Entrenador | Date | X |  |
| Telefono1\_ Entrenador | varchar(25) | X |  |
| Telefono2\_ Entrenador | varchar(25) |  |  |
| Email1\_ Entrenador | varchar(30) | X |  |
| Email2\_ Entrenador | varchar(30) |  |  |
| Distrito\_ Entrenador | varchar(30) | X |  |
| Canton\_ Entrenador | varchar(30) | X |  |
| Provincia\_ Entrenador | varchar(10) | X | Valores: San José, Alajuela, Cartago, Heredia, Guanacaste, Puntarenas, Limón |
| Pais\_ Entrenador | varchar(20) | X |  |
| Direccion\_ Entrenador | varchar(150) | X |  |
| Cuenta\_Bancaria\_ Entrenador | varchar(20) | X |  |
| Activo\_ Entrenador | bit | x |  |

|  |  |  |  |
| --- | --- | --- | --- |
| VACACIONES \_INCAPACIDADES\_EVENTOS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Vac\_Inc\_Evento | int | x | PK, IDENTITY |
| Cod\_Entrenador | int | X | FK |
| Tipo\_Evento | Varchar(30) | x | 'VACACIONES','INCAPACIDAD','PERMISO','LICENCIA','OTROS' |
| Fecha\_Inicio | date | X | >= fecha actual |
| Fecha\_Fin | date | x | >= fecha actual |
| Detalle\_Evento | Varchar(150) |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| REGISTRO\_HORAS\_LABORADAS\_ENTRENADOR | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Registro\_Horas\_Laboradas | int | x | PK, IDENTITY |
| Cod\_Entrenador | int | X | FK |
| Fecha\_Laborada | date | X | >= fecha actual |
| Hora\_Ingreso\_Laborada | Time(0) | X |  |
| Hora\_Salida\_Laborada | Time(0) | X |  |
| Detalle\_Laborado | varchar(150) |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| HORARIOS\_ENTRENADORES | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Horario\_Entrenador | int | x | PK, IDENTITY |
| L\_Hor | bit | x |  |
| k\_Hor | bit | X |  |
| M\_Hor | bit | X |  |
| J\_Hor | bit | x |  |
| V\_Hor | bit | x |  |
| S\_Hor | bit | x |  |
| D\_Hor | bit | x |  |
| Hora\_Inicio\_Hor | Time(0) | x |  |
| Hora\_Fin\_Hor | Time(0) | x |  |
| Descripcion\_Hor | Varchar(200) | x |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ATLETAS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Atleta | int | x | PK, IDENTITY |
| ID\_ Atleta | varchar(30) | X | UNIQUE |
| Nombre\_ Atleta | varchar(20) | X |  |
| Apellido1\_ Atleta | varchar(20) | X |  |
| Apellido2\_ Atleta | varchar(20) | X |  |
| Nacimiento\_ Atleta | Date | X |  |
| Telefono1\_ Atleta | varchar(25) | X |  |
| Telefono2\_ Atleta | varchar(25) |  |  |
| Email1\_ Atleta | varchar(30) | X |  |
| Email2\_ Atleta | varchar(30) |  |  |
| Distrito\_ Atleta | varchar(30) | X |  |
| Canton\_ Atleta | varchar(30) | X |  |
| Provincia\_ Atleta | varchar(10) | X | Valores: San José, Alajuela, Cartago, Heredia, Guanacaste, Puntarenas, Limón |
| Pais\_ Atleta | varchar(20) | X |  |
| Direccion\_ Atleta | varchar(150) | x |  |
| Observacion\_ Atleta | Varchar(150) |  |  |
| Activo\_ Atleta | bit | x |  |

|  |  |  |  |
| --- | --- | --- | --- |
| MATRICULAS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Matricula | int | X | PK, IDENTITY |
| Cod\_Atleta | int | x | FK |
| Fecha\_Matricula | Date | X | >= fecha actual |
| Estado\_Matricula | Varchar(15) | X | Valores: en curso, abandono, aprobado, reprobado |
| Monto\_Cancelado\_Matricula | Decimal(10,2) | X | Def: 0 |
| Tipo\_Cobro\_Matricula | Varchar(20) | X | Valores: curso, curso y matricula |
| Tipo\_Pago\_Matricula | Varchar(15) | x | Valores: transferencia, sinpe, tarjeta, efectivo, otro |
| Detalle\_Matricula | Varchar(150) |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| MODULOS\_ABIERTOS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Modulo\_Abierto | int | x | PK, IDENTITY |
| Cod\_Entrenador | int | x | FK |
| Cod\_Modulo | Int | X | FK |
| Cod\_Horario\_Modulo | Varchar(50) | X | FK |
| Costo\_Modulo | decimal(10,2) | X | >= 0 |
| Inicio\_Modulo\_A | Date | X | >= fecha actual |
| Fin\_Mod\_A | date |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| MODULOS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Modulo | Int | X | PK, IDENTITY |
| Nombre\_Modulo | Varchar(15) | X |  |
| Duracion\_Modulo | decimal(10,2) | X |  |
| Requisitos\_Modulo | Varchar(150) | X |  |
| Detalle\_Modulo | Varchar(150) | X |  |

|  |  |  |  |
| --- | --- | --- | --- |
| MODULOS\_PROGRAMAS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Modulo\_Programa | int | x | PK, IDENTITY |
| Cod\_Modulo | Int | X | FK |
| Cod\_Programa | int | X | FK |

|  |  |  |  |
| --- | --- | --- | --- |
| PROGRAMAS | | | |
| **Campo** | **Tipo** | **Not Null** | **Observaciones / Restricciones** |
| Cod\_Programa | Int | X | PK, IDENTITY |
| Nombre\_Programa | Varchar(15) | X |  |
| Duracion\_Programa | int | x |  |
| Descripcion\_Programa | Varchar(150) | X |  |
| Requisito\_Programa | Varchar(150) | X |  |
| Observaciones\_Programa | Varchar(150) | X |  |
| Adicional\_Programa | Varchar(150) |  |  |

# Conclusiones

UML facilita el identificar y definir clases, además de tener un mejor ordenamiento al momento de crear un buen sistema, por lo cual, es un gran acierto para realizar cada uno de los temas, dando un mejor entendimiento a estos.

Lo aprendido en las últimas semanas ayudó a asimilar la importancia que los diagramas tienen en cuanto a aprender el flujo de la información, se abordó un verdadero aprendizaje para la importancia del UML y sus diferentes maneras de uso.

Se utilizó algunos aspectos realizados en el módulo de bases de datos, como el proyecto hacia el IEET, en esta ocasión se todo la base de datos y se realizará a futuro una interfaz gráfica en C#.

# Cambios para la Fase 2

* Se cambió el término “estudiante” a “atleta” y “profesor” por “entrenador” en este documento y en la base de datos.
* Se eliminó la historia de usuario que hacía uso de creación y modificación de institutos y se agregará el instituto con anterioridad.
* Se agregó la palabra “Recepcionista” a la historia de usuario, además, se dividió en dos la historia de usuario para crear una nueva, antes abarcaba el ingreso y modificación de usuarios y el proceso de matrículas.
* Se modificó toda la historia de usuario para que sea más adecuada a “Modulos\_Abiertos”.
* Los módulos, programas y horarios de los módulos abiertos serán creados con anterioridad.
* Se modificaron los requerimientos solicitados.
* Se modificaron las prioridades.

Diagrama de Clases:

* Se eliminó la relación entre entrenadores y atletas.
* Programa está compuesto de módulo.
* Matrículas ahora es una tabla de unión.
* Se aplicó el polimorfismo.

Diagrama Lógico:

* Se eliminó la tabla MODULOS\_ELEGIDOS porque no tenía uso.