

**Instituto Tecnológico de Costa Rica**

**Escuela de Ingeniería en Computación**

**Bases de Datos I**

**Proyecto**

**Integrantes**

Lermith Biarreta Portillo (2020277475)

Adrián Herrera Segura (2019081563)

**Profesor**

Allan Rodríguez Dávila

**Grupo**

60

**Fecha de Entrega**

3 de noviembre

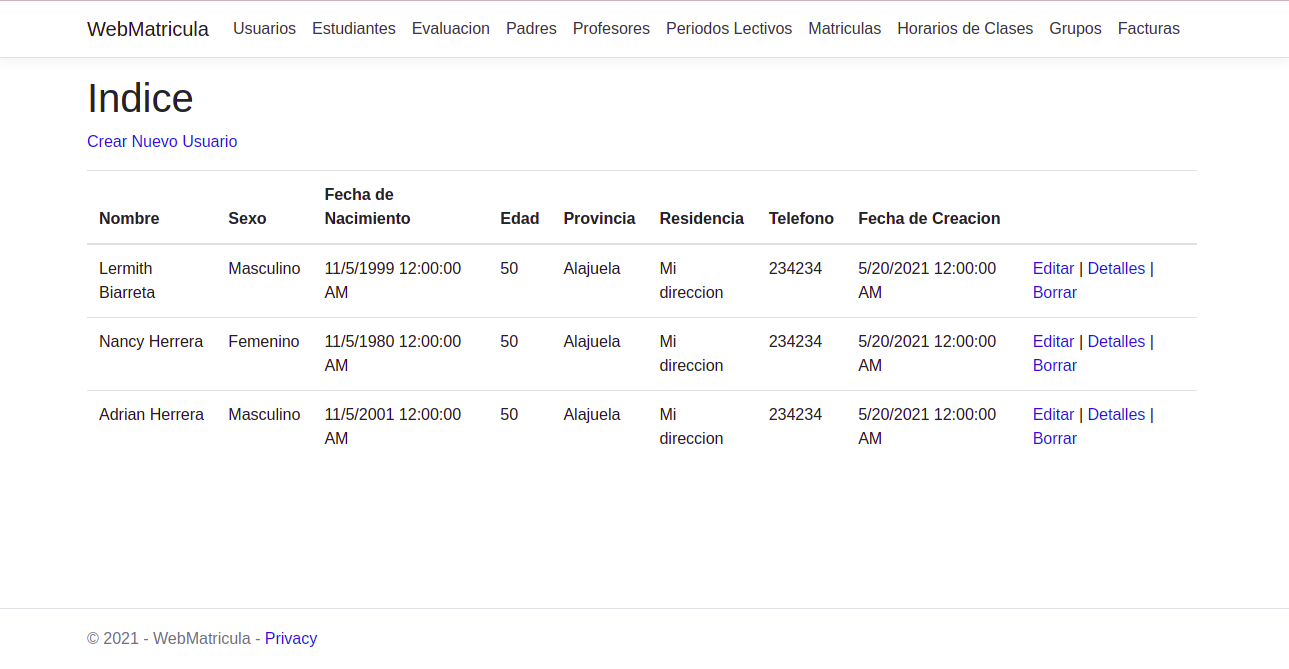
II Semestre, 2021

**Manual de Usuario**

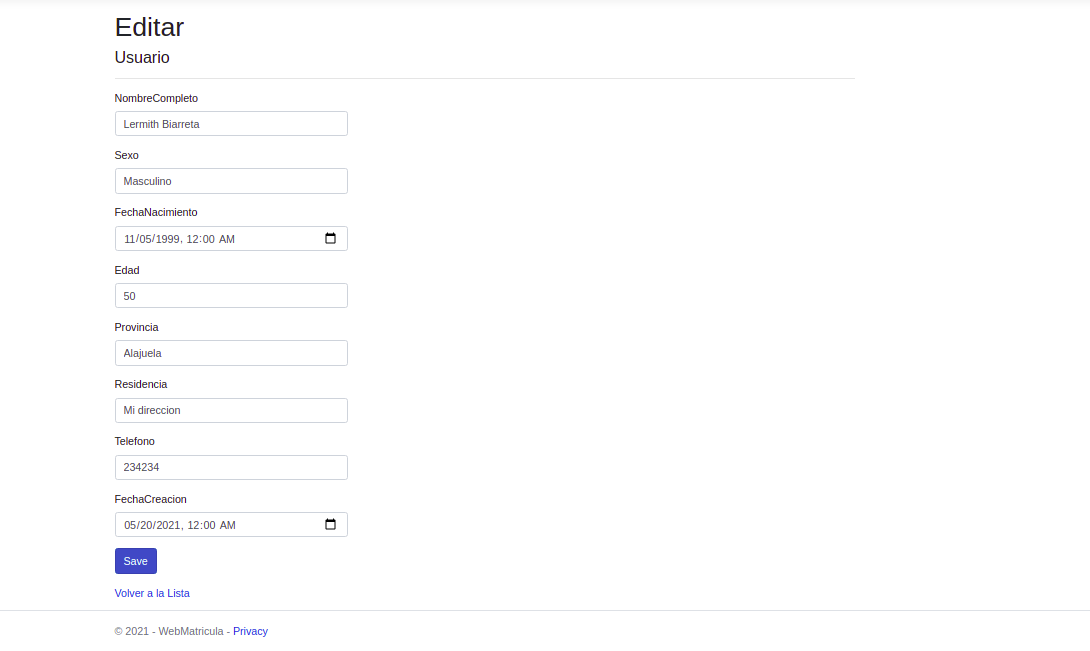
* Para el funcionamiento correcto de este programa es necesaria la instalación de React, ASP.NET Core en formato API, y Microsoft SQL Server Management Studio.
* Una vez descomprimidos el .ZIP donde se encuentran todos los archivos necesarios para el funcionamiento correcto del programa, se debe ejecutar en ASP.NET Core y React.

**Instrucciones durante la ejecución del programa:**

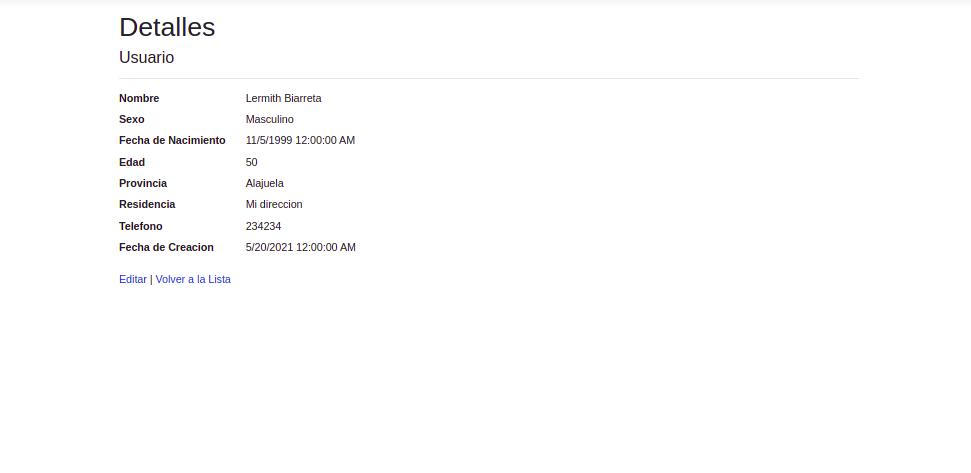
* Una vez ejecutado el programa, al aparecer la ventana de ejecución, aparecerá el menú principal del programa, la cual aparecerá como una barra en la parte superior.

****

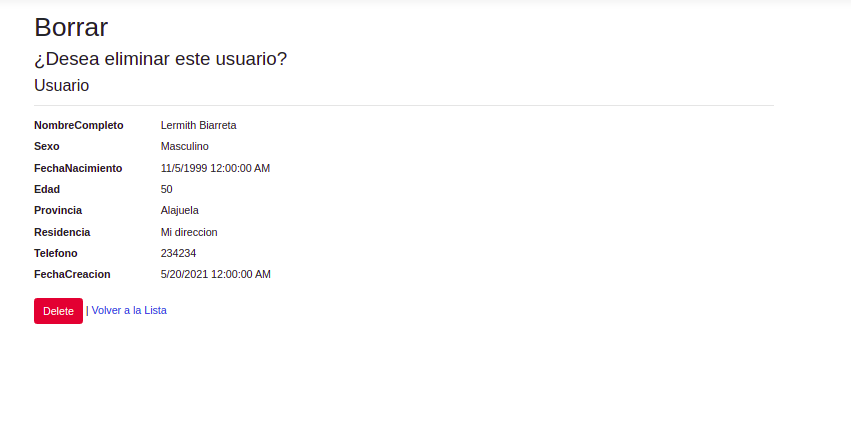
* En la ventana “Usuarios”, el posible la creación de usuarios que tendrán su respectivo papel en el centro educativo.

****

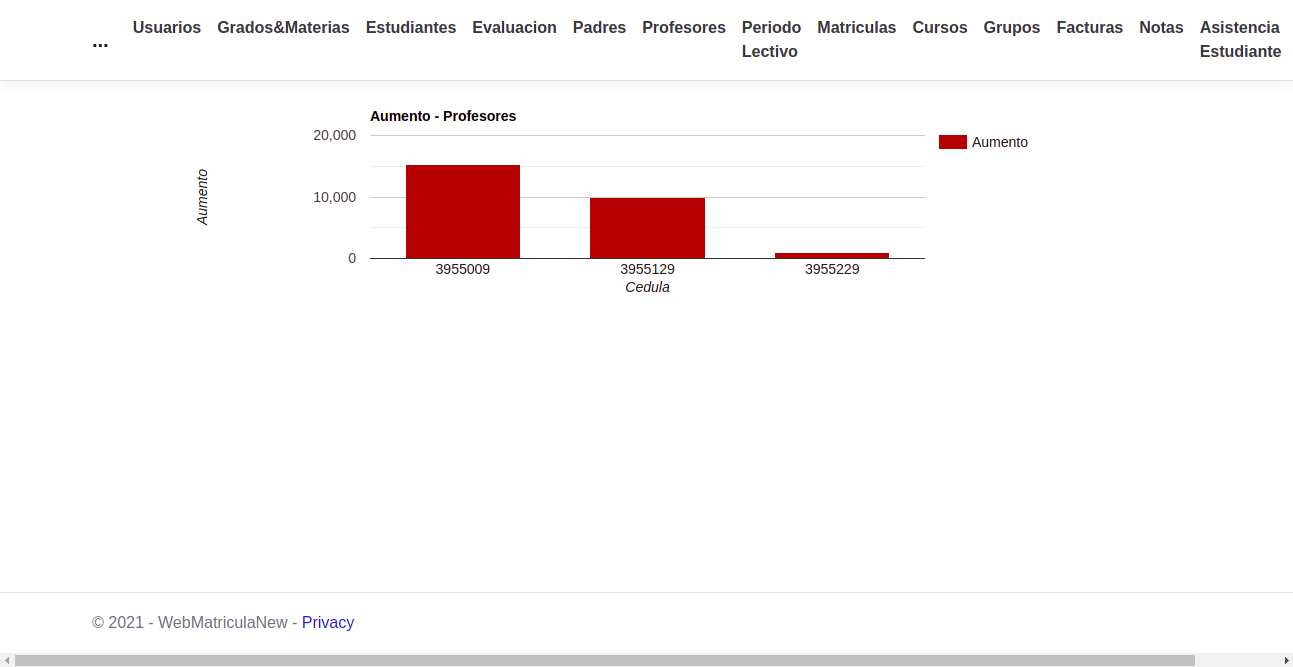
* Es posible desplegar una ventana con los detalles más explicados de cada usuario creado en el programa.

****

* Es posible eliminar los usuarios creados de la base de datos.



* El formato de inserción, listar, eliminar y actualizar información es la misma para toda la información de este estilo (estudiantes, profesores, padres, grupos, cursos, etc.)
* Hay ventanas con diferentes gráficos que muestras de forma más sencilla y atractiva los datos recopilados por la aplicación.



**Descripción del problema:**

La función principal de un sistema de gestión educativa es compilar toda la información de un sector educativo, acumulándolo en una base de datos. Normalmente se compila la información de padres, estudiantes y profesores. Esto facilita la comunicación de los integrantes del centro educativo, ya sea para ingresar calificaciones, registros de asistencia, entre otros.

El sistema debe tener un módulo para manejo de usuarios, donde se podrán crear, teniendo sus diferentes roles (estudiante, padre o profesor), modificar su información, tener acceso a sus opciones y funcionalidades dependiendo de su rol designado. Por cada usuario, se guardará su cédula, nombre completo, sexo, fecha de nacimiento, edad, provincia y lugar de residencia, teléfono y su fecha de creación.

El sistema debe tener un sistema de manejo de profesores, que permita registrar, editar, eliminar y visualizar la información de los profesores. A estos usuarios se les incluye un salario, una materia que imparten y un grupo. Tendrán acceso a una gestión de evaluaciones y un registro de asistencias.

El sistema debe tener un sistema de manejo de padres, que permita registrar, editar, eliminar y visualizar la información de los padres. A estos usuarios se les incluye una profesión, un conyugue y el teléfono del conyugue. Además, deben tener adeudados los costos de matricula y mensualidad de los hijos, y pueden tener acceso a el gestor de evaluaciones para ver el rendimiento de sus hijos.

El sistema debe tener un sistema de manejo de estudiantes, que permita registrar, editar, eliminar y visualizar la información de los estudiantes. A estos usuarios se les asignará un grado a cursar, y un periodo indicado. Además de que tendrán un gestor de evaluaciones por cada curso que lleven.

El sistema debe tener un sistema de manejo de periodos lectivos, que permita registrar, editar, eliminar y visualizar la información de los periodos lectivos. Por cada periodo se guarda el año, el número de periodo, y su fecha de inicio y final. Un periodo abierto puede cerrar, pero esto implica que todos los cursos que tenga ese periodo también cerrarán. Una vez que todos los cursos estén cerrados, se revisará si los estudiantes están aprobados o reprobados.

El sistema debe tener un sistema de manejo de grupos, que permita registrar, editar, eliminar y visualizar la información de los grupos. Para cada grupo se guardará un código de grupo, nombre, profesor a cargo, periodo y año, materia y grado al que pertenece. A cada grupo se le deben asignar estudiantes previamente matriculados y un profesor.

El sistema debe tener un sistema de manejo de matrículas, que permita registrar, editar, eliminar y visualizar la información de las matrículas. La matrícula debe indicar la cédula del estudiante y el periodo a matricular, y el sistema lo agregará a un grupo disponible. Se generarán doce cobros de mensualidad por cada estudiante.

El sistema debe tener un sistema de manejo de pagos de padres de familia, que permita registrar, editar, eliminar y visualizar la información de los pagos de padres de familia. Por cada cédula del padre, se podrá generar un historial de pagos, e indicando el total. Además de tener acceso a cobros pendientes, seleccionarlos y facturarlos. La factura debe asociarse a los cobros y guardar fecha de pago, usuario, monto total, se calcula impuesto del 2%, se debe mostrar el detalle de lo pagado.

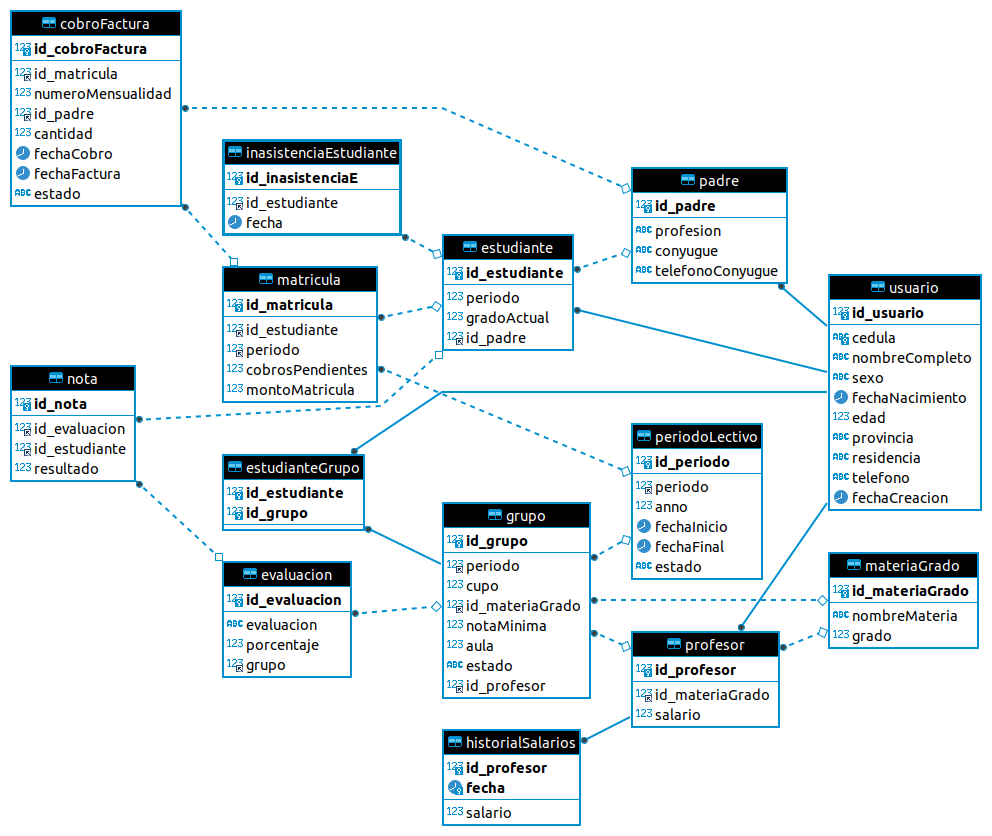
**Diseño del programa:**

Se optó por crear el diseño de la aplicación, en la parte del frontend, usando la aplicación React, siendo esta una herramienta de Java Script diseñada para crear interfaces de usuario. Se eligió esta aplicación antes que las demás que cumplen la misma función, por su sencillez a la hora de trabajar y la cantidad de ayuda en internet para tener un buen manejo de la misma.

Por otra parte, en la parte del backend, se optó por usar ASP.NET CORE, específicamente usando el framework API, ya que esta herramienta permite construir el software de una aplicación dividiendo la composición en tres partes: modelo, vista y controlador. Esta ayuda en la organización de la aplicación en la parte del software, y facilita la conexión con otras aplicaciones, como por ejemplo React para la parte del frontend.

En la parte de la base de datos, se utiliza MySQL, esto por la compatibilidad que tiene este gestor de bases de datos con el ASP.NET CORE y a su vez con React. Además de su facilidad a la hora de crear bases de datos, consultas y funciones necesarias para la correcta funcionalidad del programa.

**Diseño del diagrama relacional:**



**Librerías usadas:**

* **Microsoft.AspNetCore.App:** Framework de ASP.NET Core que contiene ensambladores desarrollados y compatibles con Microsoft.
* **Microsoft.EntityFrameworkCore.Sqlite:** Permite usar Entity Framework Core con SQLite.

* **Microsoft.EntityFrameworkCore.SqlServer:** Permite usar Entity Framework Core con Microsoft SQL Server.
* **Microsoft.EntityFrameworkCore.Tools:** Biblioteca que ayuda con las tareas de desarrollo en el campo de diseño.
* **Microsoft.VisualStudio.Web.CodeGeneration.Design:** Biblioteca que contiene comandos para generar controladores y vistas.
* **MySql.Data.EntityFrameworkCore:** Biblioteca que contiene clases para crear dominios de la base de datos para que así combinen con las clases de la estructura de programación ya creada.
* **MySqlConnector:** Soporte de conexiones de bases de datos SQL con diferentes aplicaciones .NET.
* **Pomelo.EntityFrameworkCore.MySql:**  Proveedor de Entity Framework Core que permite el uso del mismo con bases de datos SQL.

**Análisis de resultados:**

**Objetivos alcanzados:**

* La creación de usuarios y la asignación de los roles se crea de manera exitosa.
* La creación de cursos y la asignación de estudiantes a los cursos creados, junto con su profesor se crea de manera exitosa.
* Que cada estudiante tenga su evaluación dependiendo de cada curso funciona de forma exitosa.
* El sistema de matricula y mensualidad del centro educativo funciona de manera exitosa.
* El sistema de asistencia funciona de manera exitosa.

**Objetivos no alcanzados:**

* La consulta de cobros por padre no se logró representar de manera correcta.
* Dividir los periodos lectivos por tipo de usuario no se pudo crear de manera correcta.

**Razones por la cuales no pudieron lograrse:**

* Falta de tiempo y exceso de complejidad.

**Bitácora de GitHub:**

- commit acdb55881669d41991fb6c52075d2cb744a84fd4 (HEAD -> main, origin/main, origin/HEAD)

Author: LermithB <lermith17@gmail.com>

Date: Wed Nov 24 01:36:51 2021 -0600

3er Proyecto de Bases de Datos: Lermith Biarreta, Adrian Herrera

- commit 1804795f415c434464424b12d0bf350dd7f1fa1e

Author: LermithB <lermith17@gmail.com>

Date: Wed Nov 24 01:31:59 2021 -0600

3er Proyecto de Bases de Datos: Lermith Biarreta, Adrian Herrara

- commit d3835d5a56774c69dbc208949881c48e949bf997

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:46:19 2021 -0600

Documentacion: Segunda Entrega

- commit 42f2c698e2c195b6f365e2b9d23c373db02c22df

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:40:26 2021 -0600

- commit acdb55881669d41991fb6c52075d2cb744a84fd4 (HEAD -> main, origin/main, origin/HEAD)

Author: LermithB <lermith17@gmail.com>

Date: Wed Nov 24 01:36:51 2021 -0600

3er Proyecto de Bases de Datos: Lermith Biarreta, Adrian Herrera

- commit 1804795f415c434464424b12d0bf350dd7f1fa1e

Author: LermithB <lermith17@gmail.com>

Date: Wed Nov 24 01:31:59 2021 -0600

3er Proyecto de Bases de Datos: Lermith Biarreta, Adrian Herrara

- commit d3835d5a56774c69dbc208949881c48e949bf997

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:46:19 2021 -0600

Documentacion: Segunda Entrega

- commit 42f2c698e2c195b6f365e2b9d23c373db02c22df

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:40:26 2021 -0600

Factura Original Implementada

- commit 345ba0ba8d900de85c1a149abca9429779dc0d1d

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:32:19 2021 -0600

Generacion de CRUD, factura implementada, base de datos actualizada, entrega Domingo 7/11

- commit 39e26f941484b1ac2d8b854a047739d231e07807

Author: LermithB <lermith17@gmail.com>

Date: Sun Oct 31 18:41:25 2021 -0600

Script Actualizado

- commit eaac9c4e7535af4b4ff4b085e74d792fa9e13d0a

Author: LermithB <lermith17@gmail.com>

Date: Wed Oct 27 21:32:11 2021 -0600

- commit acdb55881669d41991fb6c52075d2cb744a84fd4 (HEAD -> main, origin/main, origin/HEAD)

Author: LermithB <lermith17@gmail.com>

Date: Wed Nov 24 01:36:51 2021 -0600

3er Proyecto de Bases de Datos: Lermith Biarreta, Adrian Herrera

- commit 1804795f415c434464424b12d0bf350dd7f1fa1e

Author: LermithB <lermith17@gmail.com>

Date: Wed Nov 24 01:31:59 2021 -0600

3er Proyecto de Bases de Datos: Lermith Biarreta, Adrian Herrara

- commit d3835d5a56774c69dbc208949881c48e949bf997

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:46:19 2021 -0600

Documentacion: Segunda Entrega

- commit 42f2c698e2c195b6f365e2b9d23c373db02c22df

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:40:26 2021 -0600

Factura Original Implementada

- commit 345ba0ba8d900de85c1a149abca9429779dc0d1d

Author: LermithB <lermith17@gmail.com>

Date: Sun Nov 7 23:32:19 2021 -0600

Generacion de CRUD, factura implementada, base de datos actualizada, entrega Domingo 7/11

- commit 39e26f941484b1ac2d8b854a047739d231e07807

Author: LermithB <lermith17@gmail.com>

Date: Sun Oct 31 18:41:25 2021 -0600

Script Actualizado