

ACTIVIDAD EVALUATIVA M3 (EPE 2)



Yonathan Francisco Ancheo González
Taller de Aplicaciones Punto Net.

Tabla de contenido

1. Introducción	3
2. Objetivos	4
3.1. Breve descripción de Visual Studio 2019 y Visual Basic con POO.	5
3.2 Breve descripción de MySql y Workbench 8.0 CE.	6
4. Casos de uso.....	7
4.2 Casos de Uso.....	8
4.2.1 Listar Atenciones	8
4.2.2 Registrar una nueva atención	9
4.2.3 Consideraciones.....	9
4.3 Diagrama de flujo.	11
5.1. Creación de la base de datos.	12
5.2 Desarrollo de la base de datos en MySQL Workbench 8.0 CE.	13
6. Desarrollo de la aplicación.	14
6.1 Descripción del Desarrollo	14
6.1.1 Arquitectura del Sistema.	14
6.1.2 Implementación de Funcionalidades Principales:	14
6.1.3 Integración con la Base de Datos.	15
6.1.4 Decisiones de Diseño y Implementación:	15
7. Desarrollo de la interfaz.	16
1. Form.....	16
2. Labels:.....	16
3. TextBox	16
4. ComboBox:.....	17
8. Creación de clases de la Persistencia.	18
8.1 Clase conexión.....	18
8.1.1 Atributo con:	18
8.1.2 Método conectar():	18
8.1.3 Método desconectar():.....	18
8.2 Clase PAtenciones	19
8.2.2 Constructores:.....	19

8.2.4 Propiedades (Properties):.....	19
Creación de clases para la capa Lógica	20
9.1 Herencia y atributos:	20
9.2 Constructores:	20
9.3 Propiedades (Properties):	20
10. Creación para la vista y eventos.....	21
10.1 Métodos de eventos de los controles:	21
11. Pruebas	23
11.1 Verificando que nuestra consulta SQL consolide la información solicitada.	23
11.2 Corremos la aplicación desde VS 2019 para corroborar una correcta visualización de los elementos utilizados.	24
12.3 Pulsando el botón “Listar Atenciones” para mostrar la información.	25
12.4 Cargar datos de cliente.	25
12.5 Ingresar una Atención.....	26
13.1 Conclusiones	28
13.2 Mejoras futuras para la aplicación.	28
14. Repositorio epe2_consulta_puntonet	29

1. Introducción

La programación orientada a objetos (POO) representa un paradigma fundamental en el desarrollo de software moderno, proporcionando una metodología poderosa para organizar y estructurar el código. Al centrarse en la representación de entidades como objetos que encapsulan datos y comportamientos, la POO fomenta la reutilización, la modularidad y la mantenibilidad del código. Comprender y aplicar los principios de la POO no solo mejora la eficiencia del desarrollo, sino que también facilita la escalabilidad de los proyectos, permitiendo a los desarrolladores construir sistemas complejos de manera más estructurada y comprensible.

En el contexto de la integración con bases de datos, conocer los métodos para establecer conexiones y realizar operaciones CRUD (Create, Read, Update, Delete) es crucial. Esta habilidad no solo permite a los desarrolladores interactuar dinámicamente con datos persistentes, sino que también facilita la creación de aplicaciones robustas y funcionales que responden a las necesidades de los usuarios y las organizaciones.

El presente proyecto tiene como objetivo desarrollar una aplicación de gestión de atenciones médicas para una institución de salud específica. La institución en cuestión es un centro médico que brinda servicios de atención médica en un área de atención cerrada, lo que implica un entorno donde los pacientes reciben cuidados continuos durante un período de tiempo determinado.

Se ha decidido desarrollar una aplicación que permita el registro de atenciones realizadas a los pacientes, teniendo un el personal medico con sus respectivas áreas. Visual Basic ha sido seleccionado como el lenguaje de programación principal debido a su robustez, portabilidad y amplio uso en aplicaciones empresariales. El entorno de desarrollo Visual Basic será utilizado para facilitar el proceso de desarrollo, aprovechando su conjunto completo de herramientas y funcionalidades avanzadas. MySql será nuestro gestor de base de datos para la creación de esta, ya que es unos de los gestores bastantes utilizados en el ambiente laboral

El proyecto realizado para esta evaluación se encuentra en mi repositorio de GitHub, el cual se encuentra en el siguiente enlace:

[https://github.com/YonaAncheo/02 Consulta Medica VB/tree/main/01_epe2_consulta_puntonet](https://github.com/YonaAncheo/02_Consulta_Medica_VB/tree/main/01_epe2_consulta_puntonet)

2. Objetivos

2.1 Dominar los conceptos fundamentales de la programación orientada a objetos, como clases, objetos, herencia, polimorfismo y encapsulamiento.

2.2 Aplicar los principios de diseño orientado a objetos para modelar sistemas complejos de manera efectiva y eficiente.

2.3 Familiarizarse con Visual Studio 2019 como entorno de desarrollo integrado (IDE) y utilizar Visual Basic como lenguaje de programación para implementar aplicaciones orientadas a objetos.

2.4 Aprender a establecer conexiones a una base de datos MySQL desde aplicaciones Visual Basic utilizando métodos seguros y eficientes, como ADO.NET.

2.5 Desarrollar habilidades en la creación de consultas SQL, manejo de transacciones y gestión de datos para realizar operaciones CRUD de manera efectiva.

2.6 Desarrollar una aplicación de gestión de atenciones médicas: El objetivo principal es crear una aplicación que permita registrar, dar seguimiento y gestionar las atenciones médicas brindadas a los pacientes en el área de atención cerrada de la institución de salud.

2.7 Optimizar los procesos de registro y seguimiento: Se busca mejorar la eficiencia en los procesos de registro y seguimiento de las atenciones médicas para garantizar un flujo de trabajo más fluido y una mejor organización de la información.

2.8. Preparar el terreno para futuras mejoras: Se debe diseñar la aplicación de manera que sea escalable y fácil de mantener, permitiendo la incorporación de nuevas funcionalidades y mejoras en el futuro.

3.1. Breve descripción de Visual Studio 2019 y Visual Basic con POO.

Visual Studio 2019 es una potente herramienta de desarrollo integrado (IDE) que proporciona un entorno completo para escribir, depurar, compilar y ejecutar aplicaciones. Cuando se combina con el lenguaje de programación Visual Basic, especialmente en el contexto de la programación orientada a objetos (POO), ofrece varias características y ventajas.

A continuación, describo cómo sería el uso de Visual Studio 2019 junto con Visual Basic en un enfoque de programación orientada a objetos.

3.1 Creación de un Proyecto

3.1.1 Abre Visual Studio 2019 y selecciona "Crear un nuevo proyecto".

3.1.2 Selecciona "Visual Basic" en la lista de plantillas y elige el tipo de proyecto que deseas crear (por ejemplo, Windows Forms Application, Class Library, etc.).

3.1.3 Definir el nombre y la ubicación del proyecto y haz clic en "Crear" para generar la estructura inicial del proyecto.

3.2 Diseño de Clases

3.2.1 Comenzamos definiendo las clases que vamos a utilizar, para esto tendremos la clase principal asociada al formulario, el cual llamaremos "Calculadora_form". Continuamos definiendo fuera de la clase principal la que utilizaremos para las operaciones matemáticas, a esta clase la llamaremos "Operaciones".

3.2.2 En la clase "Operaciones" definiremos las funciones suma, resta, multiplicación y división.

3.2.3 En cuanto a la clase principal "Calculadora_form", contendrá las funciones y métodos setter y getter, para establecer y obtener la información de los campos de texto. Agregaremos también un método para limpiar los campos de texto y dejarlos en su estado inicial, para esta evaluación será "0". Además, cualquier otra función que se requiera para el control de errores.

3.3 Depuración y Pruebas

3.3.1 Utilizar las herramientas de depuración de Visual Studio (puntos de interrupción, seguimiento de variables, etc.) para identificar y corregir errores en tu código.

3.3.2 Ejecutar pruebas unitarias para verificar el funcionamiento de tus clases y métodos.

3.3.3 Utilizar las opciones de ejecución y perfilado para evaluar el rendimiento de tu aplicación.

3.2 Breve descripción de MySql y Workbench 8.0 CE.

MySQL es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto ampliamente utilizado en aplicaciones web y empresariales. Es conocido por su fiabilidad, rendimiento y escalabilidad, y es compatible con una amplia variedad de plataformas, lo que lo convierte en una opción popular para desarrolladores y organizaciones de todos los tamaños. Algunas de sus características clave incluyen la capacidad de manejar grandes volúmenes de datos, soporte para múltiples tipos de datos y una amplia gama de funciones de consulta y administración.

MySQL Workbench, por otro lado, es una herramienta gráfica de diseño, desarrollo y administración de bases de datos MySQL. Proporciona una interfaz intuitiva que permite a los usuarios diseñar esquemas de base de datos, escribir consultas SQL, gestionar usuarios y permisos, realizar copias de seguridad y restauraciones, y mucho más. Con su funcionalidad de modelado visual, depuración de consultas SQL y capacidad de administración remota de servidores MySQL, Workbench es una herramienta valiosa para desarrolladores y administradores de bases de datos que trabajan con MySQL.

4. Casos de uso.

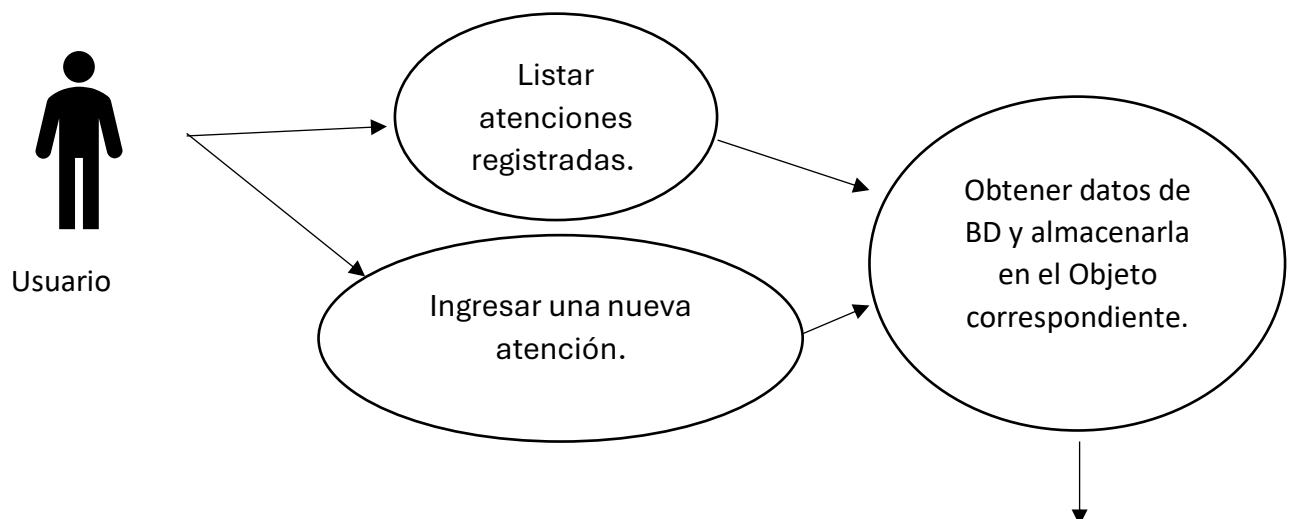
4.1 Diagrama de casos de uso.

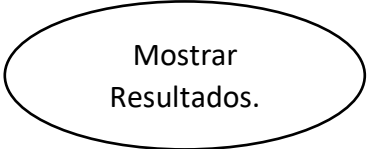
Este diagrama y los casos de uso proporcionan una visión general de las funcionalidades básicas que el sistema debe cubrir para manejar eficientemente el registro y consulta de atenciones médicas en una institución de salud.

En esta sección se describen los casos de uso principales de la aplicación. Esto incluye acciones como registrar una nueva atención médica, actualizar la información de una atención existente, eliminar una atención, etc. Cada caso de uso se detalla incluyendo los actores involucrados, las acciones que se realizan y los resultados esperados.

Para la utilización de esta aplicación, comenzamos desde el supuesto que los pacientes, especialidades médicas, médicos y las actividades ya se encuentran registradas en el sistema, por lo que no será posible ingresar una atención a un paciente no registrado.

4.1 A continuación, se presenta un diagrama de caso de uso que ilustra cómo interactúa usuario con la aplicación:





```
graph TD; UC1([Mostrar Resultados.]);
```

Mostrar
Resultados.

4.2 Casos de Uso.

4.2.1 Listar Atenciones

Descripción: Permite al usuario listar todas las atenciones registradas en el sistema.

Actores: Recepcionista, Doctor

Flujo Principal:

- El usuario selecciona la opción de listar atenciones.
- El sistema muestra una lista de todas las atenciones registradas, incluyendo información relevante como paciente, médico, especialidad y actividad clínica.

Precondiciones: Existen atenciones registradas en el sistema.

Resultado: Se muestra al usuario la lista de atenciones.

4.2.2 Registrar una nueva atención

Descripción: Permite al usuario registrar una nueva atención en el sistema.

Actores: Recepcionista, Doctor

Flujo Principal:

- El usuario ingresa el RUT (ID) asociado al paciente.
- Se muestra en pantalla el ID del paciente.
- El usuario selecciona el área correspondiente a la atención solicitada por el paciente dentro de la lista mostrada.
- El usuario selecciona al médico que corresponde dentro de la lista.
- El usuario selecciona la actividad dentro de la lista que se muestra.
- El usuario ingresa la fecha correspondiente a la del ingreso de la atención.
- El sistema valida y registra la nueva atención en la base de datos.

Precondiciones: Paciente, médico, especialidad y actividad clínica deben estar previamente registrados en el sistema.

Postcondiciones: Se registra la nueva atención en el sistema.

Flujo alternativo:

- Si el formulario no se encuentra completado, el programa muestra el mensaje en una ventana emergente con el texto “Los campos de texto y selección multiple no pueden estar vacios.”
- Si el paciente no se encuentra registrado, el programa lo indicará a través una ventana emergente, por lo que no será posible realizar la atención hasta su registro.

4.2.3 Consideraciones.

Recepcionista: Puede listar y registrar atenciones.

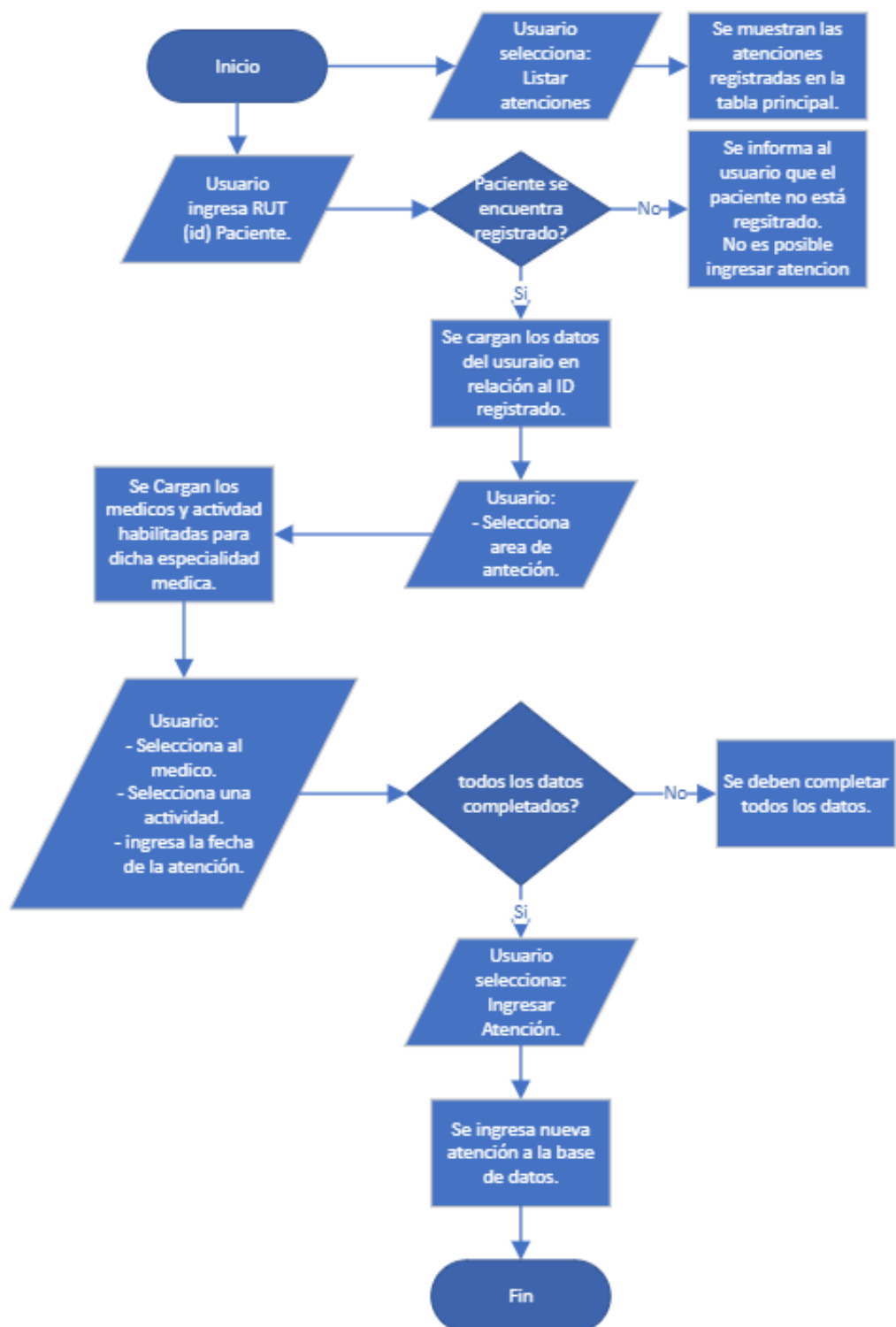
Doctor: Puede listar y registrar atenciones.

- Los casos de uso asumen que los actores ya tienen acceso al sistema y están autenticados.

- Para completar el diseño del sistema, sería importante especificar más detalles sobre los campos específicos de cada caso de uso y las validaciones necesarias para garantizar la integridad de los datos.

4.3 Diagrama de flujo.

En esta sección se incluye un diagrama de flujo que muestren el proceso que realiza el usuario para obtener la información en la aplicación y listar en la vista de tablas.



5.1. Creación de la base de datos.

En esta sección se detalla las tablas y sus relaciones diseñada para desarrollar una base de datos funcional en MySQL Workbench 8.0 CE que cumpla con los requisitos especificados para el sistema de gestión de atención médica.

Para complementar se ha generado un archivo adicional con la información de las consultas realizadas y el esquema generado. El archivo se encuentra en el repositorio de este proyecto en GitHub. A continuación, se deja el enlace al archivo:

https://github.com/YonaAncheo/02_Consulta_Medica_VB/tree/main/01_epe2_consulta_puntonet/Base%20de%20datos

5.1.1 Tabla pacientes: Esta tabla almacena la información básica de los pacientes, como su identificador único (id_paciente), nombre, edad, dirección y telefono. La clave primaria es el id_paciente.

5.1.2 Tabla especialidades: Aquí se registran las distintas especialidades médicas disponibles. Cada especialidad tiene un identificador único (Id_especialidad) y un nombre descriptivo. El Id_especialidad es la clave primaria.

5.1.3 Tabla medicos: Esta tabla contiene los datos de los médicos que trabajan en el centro médico. Cada médico tiene un identificador único (id_med), un nombre y una especialidad médica a la que pertenece, representada por el id_especialidad que es una clave foránea que se relaciona con la tabla especialidades_medicas.

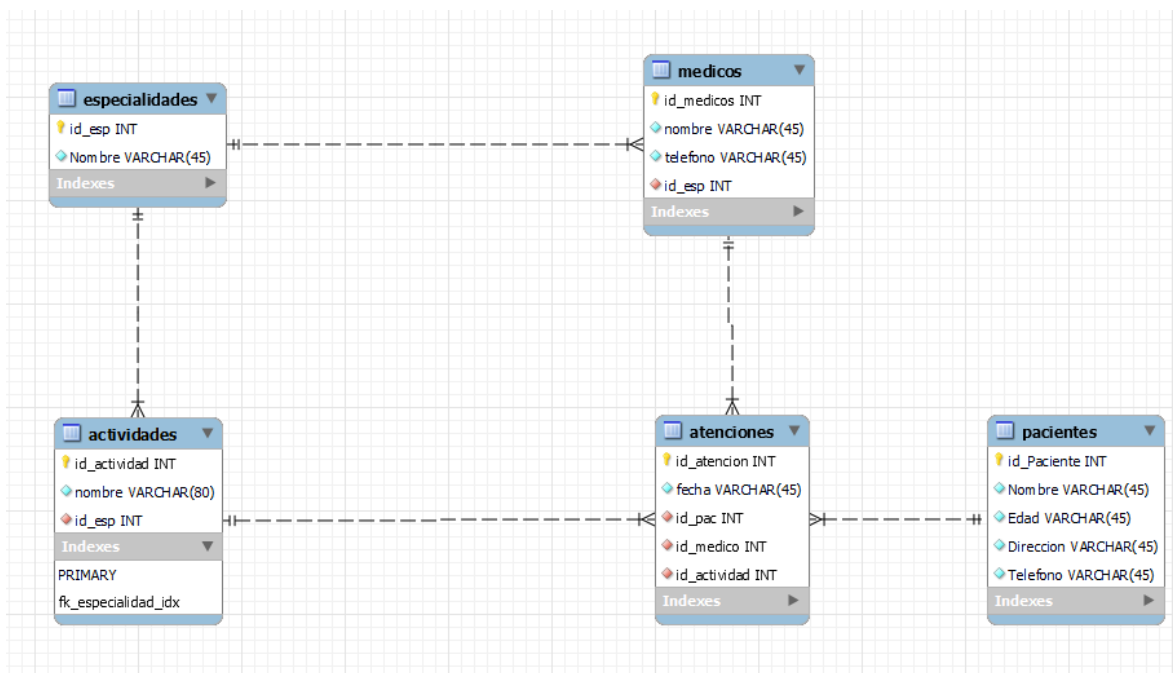
5.1.4 Tabla actividad: esta tabla contiene los datos de las actividad pertenecientes al área asociada a la especialidad. Cada actividad tiene un identificador (id_actividad), un nombre y un identificador que indica a la especialidad que pertenece.

5.1.5 Tabla atenciones: Aquí se registran las atenciones realizadas a los pacientes por los médicos. Cada atencion tiene un identificador único (id_atencion), una fecha de ingreso, y un identificador para el medico, especialidad y actividad correspondiente. Además, se registra el paciente (id_paciente) y el médico (id_med) que realizó el diagnóstico. Ambos campos son claves foráneas que se relacionan con las tablas de pacientes y médicos, respectivamente.

5.2 Desarrollo de la base de datos en MySQL Workbench 8.0 CE.

1. Abrir MySQL Workbench y crear una nueva conexión a la base de datos.
2. Una vez conectado al servidor, crear un nuevo modelo de base de datos.
3. Dentro del modelo, agregar las tablas mencionadas anteriormente y definir sus columnas, tipos de datos y restricciones de clave primaria y foránea según lo descrito en la especificación.
4. Establecer las relaciones entre las tablas utilizando la herramienta de diseño visual de Workbench. Por ejemplo, establecer una relación de clave foránea entre las tablas medicos y especialidades_medicas usando el id_especialidad como clave foránea en la tabla médicos.
5. Una vez definido todas las tablas y relaciones, generar el script SQL correspondiente utilizando la opción de ingeniería inversa de Workbench.
6. Ejecutar el script SQL generado en el servidor de base de datos para crear la estructura de la base de datos y las relaciones entre las tablas.
7. Con la tablas creadas, cargar los datos necesarios en las tablas utilizando instrucciones SQL INSERT. Este paso se realiza para obtener datos por defecto en la aplicación.
8. Finalmente, utilizamos nuevamente la ingeniería inversa para obtener la visualización del esquema generado.

Para esta evaluación, el esquema generado se puede ver en la siguiente imagen:



6. Desarrollo de la aplicación.

Esta sección es donde detallas cómo se desarrolló la aplicación. Dividiremos esta sección en subsecciones, para indicar la arquitectura del sistema, la implementación de las funcionalidades principales, la integración con la base de datos, la interfaz de usuario y las clases correspondientes para controlar la vista, lógica y persistencia de la aplicación.

Todo el código realizado para esta aplicación se encuentra en mi repositorio de GitHub, en el enlace indicado en la introducción de este documento.

6.1 Descripción del Desarrollo

6.1.1 Arquitectura del Sistema.

Para la arquitectura del sistema, se optó por seguir un enfoque de tres capas: presentación, lógica de negocio y acceso a datos. Se utilizó el patrón de diseño Modelo-Vista-Controlador (MVC) para separar claramente la lógica de presentación de la lógica de negocio y el acceso a datos. Esto proporcionó una estructura organizada y modular que facilitó el desarrollo y la mantenibilidad del código.

6.1.2 Implementación de Funcionalidades Principales:

1. Registro de atenciones médicas: Se desarrolló un formulario para registrar nuevas atenciones médicas, que incluía la selección de paciente, médico, especialidad médica, diagnóstico, fecha de ingreso y fecha de alta. Se implementaron validaciones adicionales para asegurar que los campos obligatorios fueran completados y que las fechas ingresadas fueran coherentes.
2. Listar atenciones médicas: Se proporcionara la opción para mostrar en la tabla principal de la aplicación los registros de las atenciones almacenadas en la base de datos.

6.1.3 Integración con la Base de Datos.

Se utilizó MySQL como sistema de gestión de base de datos relacional para almacenar la información de la aplicación. Se diseñó un esquema de base de datos que reflejaba la estructura de las entidades definidas en el modelo de dominio de la aplicación. Se establecieron conexiones Con MySqlConnection par visual basic para interactuar con la base de datos desde la aplicación.

6.1.4 Decisiones de Diseño y Implementación:

- Se aplicaron principios de diseño de software como la cohesión y el acoplamiento mínimo para garantizar la modularidad y la escalabilidad del sistema.
- Se utilizaron clases y métodos bien nombrados y organizados para mejorar la legibilidad y la mantenibilidad del código.
- Se implementaron mecanismos de validación tanto en el lado del cliente como en el lado del servidor para garantizar la integridad de los datos ingresados por los usuarios.
- Se realizaron pruebas unitarias para validar el comportamiento de las diferentes funciones y componentes de la aplicación, asegurando su correcto funcionamiento en diferentes escenarios.

La aplicación fue desarrollada siguiendo una arquitectura bien definida y aplicando buenas prácticas de diseño y programación para garantizar su robustez, buscando la escalabilidad en el tiempo y eficiencia.

7. Desarrollo de la interfaz.

En esta sección se detalla la interfaz gráfica de usuario (GUI) utilizando los elementos proporcionados por Visual Studio 2019 para mostrar un formulario que contenga una tabla principal para mostrar los registros, además del uso de TextBox y ComboBox, para mostrar y seleccionar la información del formulario. A continuación, se detalla el funcionamiento y la información que se muestra al usuario:

1. **Form:** Este es el contenedor y vista principal de nuestra aplicación. En el almacenaremos todos los componentes a utilizar.
2. **Labels:** He utilizado las etiquetas (label) para indicar al usuario para que corresponde cada campo.
3. **TextBox:** He utilizado los campos de texto para mostrar información y solicitar datos al usuario, para este fin, solo se habilitará el campo para el ID de la atención y el ID del usuario, los demás no estarán habilitados para el ingreso de datos.

txtIdAte	Campo de texto para el ID de la atención, habilitado para que el usuario ingrese texto
idPacienteTxt	Campo de texto para el ID del paciente, habilitado para que el usuario ingrese texto
edadCteTxt	Campo de texto para visualizar el registro edad del paciente según el ID, deshabilitado para que el usuario ingrese texto.
nomCteTxt	Campo de texto para visualizar el registro nombre del paciente según el ID, deshabilitado para que el usuario ingrese texto.
dirCteTxt	Campo de texto para visualizar el registro nombre del paciente según el ID, deshabilitado para que el usuario ingrese texto.
telCteTxt	Campo de texto para visualizar el registro nombre del paciente según el ID, deshabilitado para que el usuario ingrese texto.
txtIdEsp	Campo de texto para visualizar el ID de la especialidad según la selección de la especialidad hecha por el usuario, deshabilitado para que el usuario ingrese texto.
txtIdMed	Campo de texto para visualizar el ID del médico según la selección del médico hecho por el usuario, deshabilitado para que el usuario ingrese texto

txtIdAct	Campo de texto para visualizar el ID de la actividad según la selección de la actividad hecha por el usuario, deshabilitado para que el usuario ingrese texto
----------	---

4. **ComboBox:** En estos elementos cargaremos la información para que el usuario pueda seleccionar a través de la lista representada y no sea necesario ingresar los datos por teclado.

cmbEspecialidad	En esta lista se cargarán los datos correspondientes a las especialidades medicas registradas en la base de datos.
cmbMedico	En esta lista se cargarán los nombres de los médicos asociados a la especialidad seleccionada en cmbEspecialidades.
cmbActividad	En esta lista se cargarán los nombres de las actividades asociadas a la especialidad seleccionada en cmbEspecialidades.
cmbDia	Se genera una lista de días con números del 1 al 31
cmbMes	Se genera una lista de números para representar los meses del 1 al 12
cmbAnio	Se genera una lista de números para representar los años, se genera desde 1985 al 2024, solo como prueba para esta evaluación.

5. **Button:** Este elemento corresponde a los botones y será con los que daremos los eventos correspondientes.

datosPacBtn	Este botón contendrá el evento para cargar los datos correspondiente al paciente, de acuerdo al ID ingresado por el usuario.
ingresarBtn	Este botón contendrá el evento para ingresar una atención médica, una vez que el usuario haya realizado los pasos indicados en los casos de uso.
listarAtencionesBtn	Este botón contendrá el evento para listar las atenciones médicas.

6. **DataGridView:** Este elemento nos permite mostrar filas y columnas de datos en una cuadrícula que es personalizable. Nuestro elemento lo llamaremos dgvAtenciones.

8. Creación de clases de la Persistencia.

8.1 Clase conexión.

Esta clase en Visual Basic (.NET) llamada Conexion se encarga de manejar la conexión y desconexión a una base de datos MySQL utilizando el conector MySql.Data.MySqlClient.

8.1.1 Atributo con:

Tipo: MySqlConnection

Descripción: Representa la conexión a la base de datos MySQL.

Inicialización: Se inicializa con una nueva instancia de MySqlConnection, especificando los parámetros de conexión como servidor (server), nombre de la base de datos (database), usuario (user) y contraseña (password).

8.1.2 Método conectar():

Tipo de retorno: Boolean

Descripción: Intenta abrir la conexión con la base de datos.

Funcionalidad:

Intenta abrir la conexión utilizando con.Open().

Si la conexión se abre correctamente, devuelve True.

Si ocurre algún error durante la conexión, muestra un mensaje de error mediante MsgBox(ex.Message) y devuelve False.

8.1.3 Método desconectar():

Tipo de retorno: Sub (Procedimiento que no retorna valor)

Descripción: Cierra la conexión a la base de datos, si está abierta.

Funcionalidad:

Verifica si el estado de la conexión (con.State) es ConnectionState.Open (es decir, si la conexión está abierta).

Si la conexión está abierta, la cierra utilizando `con.Close()`.

Si ocurre algún error al intentar cerrar la conexión, muestra un mensaje de error mediante `MsgBox(ex.Message)`.

8.2 Clase PAtenciones

La clase PAtenciones en Visual Basic .NET extiende de la clase Conexion y se encarga de manejar las operaciones relacionadas con las atenciones médicas en una aplicación que utiliza una base de datos MySQL

8.2.1 Herencia y atributos:

PAtenciones hereda de la clase Conexion, lo que permite utilizar los métodos de conexión (`conectar()` y `desconectar()`) directamente en esta clase.

Atributos privados (`id_atencion`, `fecha`, `id_paciente`, `id_medico`, `id_actividad`) para almacenar los datos de una atención médica.

8.2.2 Constructores:

Sub `New()`: Constructor vacío necesario para instanciar la clase sin inicializar datos específicos inicialmente.

Sub `New(id As Integer, fech As String, pac As Integer, med As Integer, act As Integer)`: Constructor que inicializa los atributos de la atención con los valores proporcionados.

8.2.4 Propiedades (Properties):

Propiedades públicas (`Ate_Id`, `Ate_fecha`, `Ate_pac`, `Ate_Med`, `Ate_act`) que permiten acceder y modificar los atributos privados de la clase de manera controlada.

Métodos de consulta (`consultarAtenciones`, `cargarDatosPaciente`, `cargarDatosCmb`, `cargarDatosCmb2`):

`consultarAtenciones()`: Consulta todas las atenciones registradas en la base de datos, mostrando información relevante como ID de atención, fecha, nombre del paciente, nombre del médico y nombre de la actividad.

cargarDatosPaciente(id As Integer): Consulta los datos de un paciente específico según su ID.

cargarDatosCmb(tabla As String), cargarDatosCmb2(tabla As String, where As String): Cargan todos los datos de una tabla específica en la base de datos, útil para llenar controles como ComboBox (DropDownList).

Método de inserción (agregarAtencion):

Creación de clases para la capa Lógica

La clase LAtenciones en Visual Basic .NET actúa como una capa intermedia o de lógica de negocio que utiliza la clase PAtenciones para realizar operaciones relacionadas con las atenciones médicas.

9.1 Herencia y atributos:

PAtenciones hereda de la clase Conexion, lo que permite utilizar los métodos de conexión (conectar() y desconectar()) directamente en esta clase.

Atributos privados (id_atencion, fecha, id_paciente, id_medico, id_actividad) para almacenar los datos de una atención médica.

9.2 Constructores:

Sub New(): Constructor vacío necesario para instanciar la clase sin inicializar datos específicos inicialmente.

Sub New(id As Integer, fech As String, pac As Integer, med As Integer, act As Integer): Constructor que inicializa los atributos de la atención con los valores proporcionados.

9.3 Propiedades (Properties):

Propiedades públicas (Ate_Id, Ate_fecha, Ate_pac, Ate_Med, Ate_act) que permiten acceder y modificar los atributos privados de la clase de manera controlada.

Métodos de consulta (consultarAtenciones, cargarDatosPaciente, cargarDatosCmb, cargarDatosCmb2):

consultarAtenciones(): Consulta todas las atenciones registradas en la base de datos, mostrando información relevante como ID de atención, fecha, nombre del paciente, nombre del médico y nombre de la actividad.

cargarDatosPaciente(id As Integer): Consulta los datos de un paciente específico según su ID.

cargarDatosCmb(tabla As String), cargarDatosCmb2(tabla As String, where As String): Cargan todos los datos de una tabla específica en la base de datos, útil para llenar controles como ComboBox (DropDownList).

Método de inserción (agregarAtencion):

agregarAtencion(pa As PAtenciones): Inserta una nueva atención en la tabla atenciones de la base de datos, utilizando los datos proporcionados en el objeto pa.

agregarAtencion(pa As PAtenciones): Inserta una nueva atención en la tabla atenciones de la base de datos, utilizando los datos proporcionados en el objeto pa.

10. Creación para la vista y eventos.

La clase Form1 en Visual Basic .NET representa la interfaz de usuario de una aplicación destinada al registro y gestión de atenciones médicas.

10.1 Métodos de eventos de los controles:

listarAtencionesBtn_Click: Este método se ejecuta al hacer clic en el botón listarAtencionesBtn. Utiliza la clase LAtenciones para obtener todas las atenciones registradas y las muestra en un DataGridView llamado dgvAtenciones.

datosPacBtn_Click: Este método se ejecuta al hacer clic en el botón datosPacBtn. Utiliza la clase LAtenciones para cargar los datos de un paciente específico según el ID ingresado en idPacienteTxt, y muestra estos datos en los campos nomCteTxt, edadCteTxt, dirCteTxt, telCteTxt.

Form1_Load: Este método se ejecuta cuando el formulario se carga. Llama al método cmbDatos para cargar los datos de la tabla "especialidades" en el ComboBox cmbEspecialidad, y también inicializa los ComboBox para seleccionar día, mes y año.

txtIdEsp_TextChanged: Este método se ejecuta cuando cambia el texto en txtIdEsp. Llama al método cmbDatos2 para cargar los médicos y actividades asociados a la especialidad ingresada en txtIdEsp.

cmbEspecialidad_SelectedIndexChanged, cmbMedico_SelectedIndexChanged, cmbActividad_SelectedIndexChanged: Estos métodos se ejecutan cuando se selecciona un elemento en los ComboBox cmbEspecialidad, cmbMedico y cmbActividad, respectivamente. Utilizan la clase LATenciones para cargar el ID correspondiente de la especialidad, médico o actividad seleccionada y lo muestran en los campos de texto txtIdEsp, txtIdMed y txtIdAct.

ingresarBtn_Click: Este método se ejecuta al hacer clic en el botón ingresarBtn. Utiliza la clase LATenciones para agregar una nueva atención médica con los datos ingresados en los campos de texto y selección de múltiples ComboBox.

Métodos cmbDatos y cmbDatos2:

cmbDatos: Carga los datos de una tabla especificada en un ComboBox especificado (cmbDatos).

cmbDatos2: Carga los datos de una tabla especificada en un ComboBox especificado (cmbDatos), filtrando por un criterio (where).

11. Pruebas

En esta sección podemos ver algunos de las pruebas realizadas y la funcionalidad de la aplicación.

11.1 Verificando que nuestra consulta SQL consolide la información solicitada.

Query:

```
select ate.id_atencion, ate.fecha as 'Fecha', p.nombre as 'Paciente', m.nombre as 'Medico', a.nombre as 'Especialidad', a.nombre as 'Actividad'
```

```
from atenciones as ate
```

```
inner join pacientes as p on ate.id_pac=p.id_paciente
```

```
inner join medicos as m on ate.id_medico=m.id_medicos
```

```
inner join especialidades as esp on m.id_esp=esp.id_esp
```

```
inner join actividades as a on ate.id_actividad=a.id_actividad;
```

	id_atencion	Fecha	Paciente	Medico	Especialidad	Actividad
▶	1	24-06-2024	Pedro Perez	Benjamin Soto	Cirugía Laparoscópica de Vesícula Biliar	Cirugía Laparoscópica de Vesícula Biliar
	2	24-06-2024	Carlos Solar	Eduardo Cortes	Consulta de Crecimiento y Desarrollo Infantil	Consulta de Crecimiento y Desarrollo Infantil
	3	24-06-2024	Luis Rojas	Claudio Gomez M	Consulta de Gastroenterología	Consulta de Gastroenterología
	4	23-06-2024	Pedro Perez	Eduardo Cortes	Evaluación Neurológica Pediátrica	Evaluación Neurológica Pediátrica

En la imagen anterior podemos ver que la query indicada, ejecutada en Workbench, trae la información que se requiere visualizar en la aplicación sin problemas.

11.2 Corremos la aplicación desde VS 2019 para corroborar una correcta visualización de los elementos utilizados.

The screenshot shows a Windows application window titled "Form1". The form contains the following elements:

- Atención N:** A text input field.
- Datos del paciente:**
 - ID:** A text input field.
 - Cargar datos:** A button.
 - Edad:** A text input field.
 - Nombre:** A text input field.
 - Dirección:** A text input field.
 - Teléfono:** A text input field.
- Datos de la atención:**
 - Área Médica:** A dropdown menu.
 - ID:** A text input field.
 - Médico:** A dropdown menu.
 - ID:** A text input field.
 - Actividad:** A dropdown menu.
 - ID:** A text input field.
- Fecha:** Three dropdown menus for **Día** (23), **Mes** (06), and **Año** (2024).
- Atenciones de la fecha:** A button.
- Ingresar Atención:** A button.
- Listar Atenciones:** A button.
- A large, empty rectangular area at the bottom, likely intended for a table or list of appointments.

En la imagen anterior, podemos ver que la construcción de la aplicación se realiza sin problemas, pudiendo visualizar la tabla solicitada vacía, esperando a realizar el evento indicado en el botón "Listar Atenciones".

12.3 Pulsando el botón “Listar Atenciones” para mostrar la información.

Form1

Atención N:

Datos del paciente. ID Edad:

Nombre: Direccion: Telefono:

Datos de la atención:

Área Medica: ID

Médico: ID

Actividad: ID

Fecha: Dia: Mes: Año:

	id_atencion	Fecha	Paciente	Medico	Especialidad	Actividad
▶	1	24-06-2024	Pedro Perez	Benjamin Soto	Cirugía Laparosc...	Cirugía Laparosc...
	2	24-06-2024	Carlos Solar	Eduardo Cortes	Consulta de Creci...	Consulta de Creci...
	3	24-06-2024	Luis Rojas	Claudio Gomez M	Consulta de Gast...	Consulta de Gast...
	4	23-06-2024	Pedro Perez	Eduardo Cortes	Evaluación Neur...	Evaluación Neur...

En la imagen anterior, podemos ver que la aplicación muestra la información necesaria. Comprobando que la conexión a la base de datos está correcta, que los datos corresponden a los indicados en los encabezados.

12.4 Cargar datos de cliente.

Atención N:

Datos del paciente. ID Edad:

Nombre: Direccion: Telefono:

En la imagen anterior podemos ver como al ingresar el ID y presionar el botón, se cargan los datos correspondientes.

En caso el ID no esté registrado, se informa el error.

Atención N:

Datos del paciente. ID

Nombre: Dirección:

Datos de la atención:

01_epe2_consulta_puntonet

El cliente ingresado no existe.

Aceptar

12.5 Ingresar una Atención.

Atención N:

Datos del paciente. ID Edad:

Nombre: Dirección: Teléfono:

Datos de la atención:

Área Medica: ID

Médico: ID

Actividad: ID

Fecha: Día: Mes: Año:

	id_atencion	Fecha	Paciente	Medico	Especialidad	Actividad
▶	1	24-06-2024	Pedro Perez	Benjamin Soto	Cirugía Laparoscó...	Cirugía Laparoscópica de Ve...
	2	24-06-2024	Carlos Solar	Eduardo Cortes	Consulta de Creci...	Consulta de Crecimiento y De...
	3	24-06-2024	Luis Rojas	Claudio Gomez M	Consulta de Gastro...	Consulta de Gastroenterología
	4	23-06-2024	Pedro Perez	Eduardo Cortes	Evaluación Neurol...	Evaluación Neurológica Pedi...

En la imagen anterior, podemos ver como al completar los datos del formulario y presionar el botón “ingresar datos”, se informa que el ID se ha ingresado correctamente.

Luego, listando la atención podemos ver el registro generado anteriormente.

Atenciones de la fecha							Ingresar Atención	Listar Atenciones
	id_atencion	Fecha	Paciente	Medico	Especialidad	Actividad		
▶	1	24-06-2024	Pedro Perez	Benjamin Soto	Cirugía Laparoscó...	Cirugía Laparoscópica de Ve...		
	2	24-06-2024	Carlos Solar	Eduardo Cortes	Consulta de Creci...	Consulta de Crecimiento y De...		
	4	23-06-2024	Pedro Perez	Eduardo Cortes	Evaluación Neurol...	Evaluación Neurológica Pedi...		
	3	24-06-2024	Luis Rojas	Claudio Gomez M	Consulta de Gastro...	Consulta de Gastroenterología		
	5	23-06-2024	Carlos Solar	Claudio Gomez M	Consulta de Dema...	Consulta de Dermatología		

Si alguno de los campos no está completo, se genera el error mostrando en pantalla.

Atención N:

Datos del paciente. ID Cargar datos Edad:

Nombre: Direccion: Telefono:

Datos de la atención:

Área Medica: ID

Médico: ID

Actividad: ID

Fecha: Dia: Mes: Año:

Atenciones de la fecha Ingresar Atención Listar Atenciones

01_epe2_consulta_puntonet

Los campos de texto y selección multiple no pueden estar vacios.

Aceptar

13.1 Conclusiones

El informe destaca la necesidad de optimizar los procesos de registro, seguimiento y gestión de las atenciones médicas en un centro médico con atención cerrada. Para abordar esta necesidad, se propone el desarrollo de una aplicación de gestión de atenciones médicas utilizando Visual Basic como lenguaje principal y Visual Studio 2019 como entorno de desarrollo.

En esta aplicación podemos utilizar todos los conocimientos obtenidos hasta el momento la carrera correspondiente al diseño y gestión base de datos, diseño de interfaces con Java, Lenguaje de programación Java orientada a objetos para realizar todo el backend de la aplicación.

Además, se reconoce la importancia de contar con una sólida comprensión de los conceptos de diseño de sistemas, ya que estos orientan la lógica subyacente de la aplicación. La selección adecuada de una arquitectura de diseño también juega un papel fundamental en el éxito del proyecto, ya que proporciona una estructura organizada y coherente para el desarrollo de la aplicación. En este caso, la adopción del patrón Modelo-Vista-Controlador (MVC) se presenta como una elección acertada para separar las preocupaciones y facilitar el mantenimiento y la escalabilidad del sistema.

El desarrollo de este tipo de aplicaciones no solo implica la implementación de funcionalidades específicas, sino también la aplicación de un enfoque estructurado y metodológico que aproveche los conocimientos y habilidades adquiridos en el ámbito académico y profesional.

13.2 Mejoras futuras para la aplicación.

La propuesta de crear nuevas interfaces como una ventana principal tipo menú para ayudar al usuario a seleccionar la vista que requiere es una mejora muy acertada y podría agregar un valor significativo a la aplicación de gestión de atenciones médicas. En esta sección listaremos una propuesta de nuevas características que podemos implementar.

1. Menú de selección de vistas: Implementar un menú de selección de vistas en la interfaz principal permitirá al usuario navegar fácilmente entre las diferentes funcionalidades de la aplicación. Cada opción del menú podría representar una funcionalidad específica, como "Registro y actualización de paciente", "Especialidades médicas", "Actualizar datos de médico" y "Tabla de atenciones".
2. Registro y actualización de paciente: implementar un formulario interactivo para ingresar y actualizar la información del paciente, como nombre, fecha de nacimiento, entre otros datos que se puedan agregar a esta identidad.
3. Especialidades médicas: La vista de especialidades médicas podría mostrar una lista de todas las especialidades disponibles en el centro médico, junto con los médicos

que practican en cada una de ellas. Esto proporcionaría una visión general completa de la oferta de servicios médicos del centro.

4. Actualizar datos de médico: Esta funcionalidad permitiría a los usuarios actualizar la información personal y profesional de los médicos, como nombre, especialidad, horario de consulta, etc.
5. Tabla de atenciones: La vista de tabla de atenciones podría seguir siendo la funcionalidad principal de la aplicación, mostrando una lista detallada de todas las atenciones médicas registradas, con la opción de filtrar y ordenar los datos según sea necesario.
6. Mejoras en la experiencia del usuario: Se podrían implementar mejoras adicionales en la interfaz de usuario, como la inclusión de iconos descriptivos junto a cada opción del menú, una barra de búsqueda para facilitar la navegación y la incorporación de atajos de teclado para acceder rápidamente a las funcionalidades principales.

Seguridad y control de acceso: A medida que la aplicación crezca, se podría considerar la implementación de medidas de seguridad adicionales, como la autenticación de usuarios y el control de acceso basado en roles, para garantizar que solo personal autorizado pueda acceder y modificar la información sensible de los pacientes y médicos.

La creación de una interfaz principal tipo menú junto con la implementación de nuevas funcionalidades y mejoras en la experiencia del usuario pueden proporcionar una experiencia más completa y satisfactoria para los usuarios de la aplicación de gestión de atenciones médicas. Estas expansiones futuras ayudarían a hacer que la aplicación sea más eficiente, intuitiva y segura para su uso en un entorno médico.

14. Repositorio epe2_consulta_puntonet

En el siguiente enlace se encuentra el repositorio de la aplicación para visualizarla por sistema.

https://github.com/YonaAncheo/02_Consulta_Medica_VB/tree/main/01_epe2_consulta_puntonet