**COMANDO GENERAL DEL EJÉRCITO**

**ESCUELA MILITAR DE INGENIERÍA**

**“MCAL. ANTONIO JOSÉ DE SUCRE”**

**BOLIVIA**

**Imagen que contiene texto

Descripción generada automáticamentePROYECTO FINAL**

**SISTEMA DE REGISTRO DE ASISTENCIA, FALTAS Y PERMISOS DE LOS ESTUDIANTES DE LA ESCUELA MILITAR DE INGENIERÍA UNIDAD ACADEMICA COCHABAMBA**

**CRESPO ARRIARAN NATALY NICOLE C11483-9**

**CRUZ SERRANO SHARAID GABRIELA C11424-3**

**OCAMPO VALDIVIA JOAQUIN RODRIGO C11550-9**

**LLANQUE CANEDO YADIR DILAN C11881-8**

**PINTO LUJAN JHERSON ADOLFO C11546-0**

**COCHABAMBA - BOLIVIA**

**ÍNDICE**

[1 INTRODUCCIÓN 1](#_Toc184072659)

[2 ANTECEDENTES 1](#_Toc184072660)

[3 PLANTEAMIENTO DEL PROBLEMA 2](#_Toc184072661)

[3.1 IDENTIFICACIÓN DE LOS ESCENARIOS OPERACIONALES 2](#_Toc184072662)

[3.2 IDENTIFICACIÓN DEL PROBLEMA 3](#_Toc184072663)

[3.3 FORMULACIÓN DEL PROBLEMA 4](#_Toc184072664)

[4 OBJETIVOS 5](#_Toc184072665)

[4.1 OBJETIVO GENERAL 5](#_Toc184072666)

[4.2 OBJETIVOS ESPECIFICOS 6](#_Toc184072667)

[5 JUSTIFICACIÓN 6](#_Toc184072668)

[6 ALCANCE 7](#_Toc184072669)

[7 INGENIERIA DEL PROYECTO 8](#_Toc184072670)

[7.1 IDENTIFICACIÓN DE NECESIDADES 8](#_Toc184072671)

[7.2 IDENTIFICACIÓN DE LOS REQUISITOS 8](#_Toc184072672)

[7.3 CARACTERIZACIÓN DE LA SOLUCIONES 10](#_Toc184072673)

[7.4 EVALUACIÓN DE LAS SOLUCIONES 11](#_Toc184072674)

[7.5 MATRIZ DE REQUISITOS DEL SISTEMA 12](#_Toc184072675)

[7.6 DIAGRAMAS FFBD, CASOS DE USO, EDT DEL SISTEMA 13](#_Toc184072676)

[7.7 DISEÑO CONCEPTUAL DE LA BASE DE DATOS 17](#_Toc184072677)

[7.8 DISEÑO LÓGICO DE LA BASE DE DATOS 17](#_Toc184072678)

[7.9 DISEÑO FISICO DE LA BASE DE DATOS 18](#_Toc184072679)

[7.10 DML DE INSERCION DE DATOS A LAS DIFERENTES TABLAS DE LA BASE DE DATOS 19](#_Toc184072680)

[7.11 DML DE SELECCIÓN A LAS DIFERENTES TABLAS DE LA BASE DE DATOS 20](#_Toc184072681)

[7.12 DML DE ACTUALIZACIÓN A LAS DIFERENTES TABLAS DE LA BASE DE DATOS 21](#_Toc184072682)

[7.13 SELECCIÓN DE LAS ESTRUCTURAS DE DATOS DEL SISTEMA 21](#_Toc184072683)

[7.14 DESCRIPCIÓN DE NODO DE CLASES DEL SISTEMA 22](#_Toc184072684)

[7.15 RELACIÓN DE CLASE (DIAGRaMA DE CLASES UTILIZANDO UML) 25](#_Toc184072685)

[7.16 INTERFACES DE ENTRADA 27](#_Toc184072686)

[7.17 INTERFACES DE SALIDA 30](#_Toc184072687)

[7.18 CÓDIGO DEL SISTEMA (EL CÓDIGO MAS IMPORTANTE DE SUS SISTEMA 31](#_Toc184072688)

[7.19 CÁCULO DE COSTO GENERAL DEL SISTEMA DE CONTROL DE ASISTENCIA 48](#_Toc184072689)

[7.19.1 Precios del proyecto 48](#_Toc184072690)

[7.20 Participación de los elementos de la economía (Oferta y Demanda) en la prestación de servicios computacionales (Hardware y software en forma conjunta) 50](#_Toc184072691)

[7.20.1 Ecuación de la Demanda 50](#_Toc184072692)

[7.20.2 Cálculo de la elasticidad de la demanda 51](#_Toc184072693)

[7.20.3 Ecuación de la oferta 52](#_Toc184072694)

[7.21 DETERMINAR EL EQUILIBRIO MICROECONÓMICO ENTRE LOS COMPONENTES DE INGRESOS Y GASTOS 53](#_Toc184072695)

[7.21.1 Gráfica de la demanda 54](#_Toc184072696)

[7.21.2 Gráfica de la oferta 54](#_Toc184072697)

[8 CONCLUSIONES Y RECOMENDACIONES 55](#_Toc184072698)

[9 CONCLUSIONES 55](#_Toc184072699)

[10 RECOMENDACIONES 56](#_Toc184072700)

[11 Bibliografía 57](#_Toc184072701)

# INTRODUCCIÓN

La gestión de asistencias en instituciones educativas es un componente esencial para garantizar el cumplimiento de las normativas, fomentar la disciplina y medir el desempeño académico de los estudiantes. En la Escuela Militar de Ingeniería (EMI), Unidad Académica Cochabamba, este proceso se realiza actualmente de forma manual, lo que ha generado problemas como la desorganización de los datos, pérdida de información y manipulación indebida de registros. Estas deficiencias impactan negativamente en la confiabilidad del sistema y en la percepción de justicia entre estudiantes y docentes.

Este proyecto tiene como objetivo principal diseñar e implementar un sistema automatizado de registro de asistencia, faltas y permisos que modernice y optimice el proceso. Al eliminar el uso de registros en papel y centralizar la información en una base de datos segura, se espera garantizar la integridad de los datos, facilitar el acceso a los historiales por parte de los estudiantes, y mejorar la supervisión por parte de las autoridades.

# ANTECEDENTES

La EMI, reconocida por su alto nivel académico y formación integral, ha adoptado procesos tradicionales para la gestión de asistencia, confiando en registros físicos conocidos como "partes del aula". Estos son llenados manualmente por el encargado del curso y luego verificados por la Jefatura de Carrera. Sin embargo, el análisis histórico de este método ha evidenciado inconsistencias significativas. La falta de un sistema automatizado ha derivado en problemas recurrentes como pérdida de partes, dificultad para rastrear faltas recurrentes y la posibilidad de manipulación de registros sin consecuencias.

En los últimos años, las instituciones educativas han comenzado a incorporar tecnologías como bases de datos centralizadas, sistemas de autenticación biométrica y aplicaciones móviles para gestionar de manera eficiente los procesos administrativos. Inspirados en estas innovaciones, la EMI busca superar las limitaciones de su sistema actual mediante la adopción de un enfoque tecnológico que garantiza la transparencia, seguridad y accesibilidad en la gestión de asistencias. Este proyecto propone abordar estas necesidades mediante un sistema informático diseñado específicamente para las condiciones y objetivos de la institución.   
Estamos diseñando e implementando un sistema automatizado que modernice la gestión de asistencias en la EMI, eliminando el uso de registros manuales. Esto incluye el desarrollo de una base de datos centralizada, interfaces gráficas accesibles y mecanismos de control que aseguran la integridad y organización de los datos. Además, se integrará un módulo para la gestión de permisos, optimizando el flujo de información entre estudiantes, encargados y autoridades.   
El proyecto se desarrollará en fases, comenzando con la identificación de necesidades y requisitos mediante entrevistas y análisis de procesos actuales. Posteriormente, se diseñará una base de datos conceptual, lógica y física, además de interfaces gráficas intuitivas. Finalmente, se programará el sistema, se realizarán pruebas con datos reales, y se implementará un prototipo inicial en las carreras piloto de la institución, asegurando su funcionalidad y escalabilidad.

# PLANTEAMIENTO DEL PROBLEMA

## IDENTIFICACIÓN DE LOS ESCENARIOS OPERACIONALES

Tabla : Identificación de los escenarios operacionales para el registro de asistencia en la Escuela Militar de Ingeniería.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| IDENTIFICADOR PROCESO PADRE | | IDENTIFICADOR PROCESO ESPECÍFICO | NOMBRE | DESCRIPCIÓN |
| PP-01 | **Registro de asistencia** | **PE-01** | **Control de asistencia “PARTES DEL AULA”** | El registro de asistencia en el aula se rige mediante los partes diarios llenados, por el encargado de curso, el método de control es por vista o llamado de lista, tanto por el encargado como el docente correspondiente, al final de x clase, el docente como el encargado, firman respectivo parte. |
|  |  | **PE-02** | **Entrega y verificación de “PARTES DEL AULA” a Jefatura de Carrera** | El encargado de curso entrega los partes de asistencia a la Secretaría de la Jefatura de Carrera, quien se encarga de supervisar y verificar que estos documentos estén correctamente llenados. Posteriormente, se procede al archivo de los partes para su registro y seguimiento correspondiente. |
|  |  |  |  |  |
| PP-02 | **Gestión de Permisos** | **PE-03** | **Solicitud de permisos** | El estudiante realiza el permiso según orden regular:  Primeramente, dar parte al encargado de curso ya sea oficial o civil, sobre la falta de asistencia que llevara a cabo.  Llevar algún tipo de justificante a jefatura de carrera que avale que la falta fue de suma importancia.  Entrega del permiso emitido por la Jefatura de Carrera, que autoriza que la ausencia del estudiante en un determinado día sea considerada justificada. |

## IDENTIFICACIÓN DEL PROBLEMA

**Gráfico N°1:** Árbol de Problemas

Diagrama

Descripción generada automáticamente

Fuente: Elaboración propia 2024

La Escuela Militar de Ingeniería unidad académica Cochabamba, desea optimizar su sistema de registro de asistencia debido a diversas problemáticas identificadas. Se observaron incidentes frecuentes de archivado y olvido de los partes de asistencia, lo que dificulta el rastreo de faltas recurrentes y genera inconsistencias en los registros. Además, entrevistas con docentes y estudiantes revelaron un control inadecuado, con prácticas de manipulación indebida que comprometen la integridad del sistema. El análisis histórico de datos muestra una organización deficiente de la información. Entre las causas principales se destacan la desorganización y la falta de automatización, ya que el uso de partes en papel y la ausencia de un sistema automatizado permiten la pérdida y manipulación de datos. Asimismo, la complicidad y la falta de supervisión adecuada posibilitan que los registros sean manipulados sin consecuencias, mientras que el acceso limitado de los estudiantes a sus propios registros impide un monitoreo adecuado de sus faltas. Estas deficiencias generan efectos significativos, como desigualdad, ya que estudiantes que cumplen con sus obligaciones se ven perjudicados frente a quienes manipulan el sistema; una creciente desconfianza en el sistema de asistencia por parte de estudiantes, docentes y autoridades; y un impacto negativo en la disciplina y el rendimiento académico, fomentando la desmotivación y disminuyendo el desempeño estudiantil.

## FORMULACIÓN DEL PROBLEMA

Tabla 2:Conceptualización del problema principal

|  |  |
| --- | --- |
| **PROBLEMA PRINCIPAL** | ¿Cómo puede La Escuela Militar de Ingeniería resolver las deficiencias en el control de asistencias, asegurando una administración justa y precisa que elimine la manipulación indebida, que mejore la organización de los datos, y permita a los estudiantes tener acceso a su historial de asistencias? |
| **PREGUNTA CLAVE** | **OBJETIVO** |
| ¿Qué tecnologías y políticas de supervisión se deben implementar para prevenir y detectar el fraude en los registros de asistencia? | Identificar y adoptar tecnologías o sistemas de auditoría automática, que garanticen la integridad de los registros y minimicen la posibilidad de manipulación. |
| ¿Cómo puede el sistema automatizado mejorar la organización y protección de los datos de asistencia? | Desarrollar una base de datos centralizada y segura que permita un almacenamiento ordenado y un acceso rápido a la información, reduciendo el riesgo de pérdida o robo de datos. |
| ¿De qué manera se puede garantizar que los estudiantes tengan acceso a su historial de asistencias de manera segura y confiable? | Diseñar interfaces de usuario accesibles para que los estudiantes puedan consultar en tiempo real su historial de asistencia, asegurando la precisión y seguridad de los datos. |
| ¿Qué controles y penalizaciones deben establecerse para asegurar la participación en eventos obligatorios? | Implementar un sistema de registro específico para eventos obligatorios, como desfiles, que permita un seguimiento preciso y la aplicación automática de penalizaciones.  Fuente: Elaboración propia 2024 |

# OBJETIVOS

## OBJETIVO GENERAL

Desarrollar un sistema de registro de asistencia, faltas y permisos de los estudiantes de la Escuela Militar Ingeniería unidad académica Cochabamba en lenguaje de programación de C Sharp con una base de datos incluida, para brindar acceso al historial de faltas y facilitar el proceso de solicitud y autorización de permisos.

## OBJETIVOS ESPECIFICOS

* Identificar las necesidades funcionales y no funcionales del sistema actual de asistencia para determinar los requerimientos esenciales de la Escuela Militar de ingeniería.
* Desarrollar las arquitecturas funcionales y estructurales del sistema, para documentar y visualizar las operaciones del sistema.
* Crear la base de datos del sistema integrando el diseño lógico, conceptual y físico, asegurando la gestión de información de asistencia, permisos y faltas.
* Implementar sentencias SQL al sistema para la inserción, selección y actualización de datos en las tablas de la base de datos como los datos visualizados en el sistema.
* Aplicar las estructuras de datos; colas y arboles para mejorar el funcionamiento del llamado de asistencia, describirlas de manera eficiente para su implantación del programa tanto métodos y clases.
* Definir la relación entre las clases del sistema utilizando diagramas UML, estableciendo una representación clara de las interacciones y dependencias entre los diferentes componentes del sistema.
* Diseñar las interfaces gráficas del sistema tanto de entrada como de salida, priorizando la usabilidad y una estética visual atractiva para mejorar la experiencia del usuario.
* Determinar el equilibrio microeconómico entre los componentes de ingresos y gastos de una entidad económica
* Establecer la sensibilidad de la permanencia de las cantidades por adquirir o producir ante una elevación o disminución del precio de los bienes o servicios básicos

# JUSTIFICACIÓN

El desarrollo de un sistema de control de asistencias, faltas y permisos en la EMI es fundamental para modernizar la gestión administrativa y asegurar un manejo preciso de la información relacionada con la asistencia y los permisos de los estudiantes. Actualmente, los procesos manuales para registrar y gestionar estos datos provocan errores, retrasos y falta de un control centralizado, lo cual impacta negativamente en la eficiencia operativa y en la calidad del seguimiento académico. Este proyecto busca automatizar y organizar estos procesos críticos, facilitando el acceso a información actualizada y mejorando la capacidad de tomar decisiones informadas. Al incorporar herramientas de bases de datos y estructuras de datos avanzadas, el sistema permitirá gestionar grandes volúmenes de información de manera rápida y eficaz. Esto reducirá los errores administrativos, optimizará la disponibilidad de datos y contribuirá a un control más transparente y confiable, favoreciendo tanto a los estudiantes como a los administrativos de la EMI.

# ALCANCE

El proyecto tiene como objetivo principal diseñar, desarrollar e implementar un Sistema Automatizado de Gestión de Asistencias para los estudiantes de la EMI, comenzando con las carreras de Ingeniería en Sistemas e Ingeniería en sistemas Electrónicos con la posibilidad de expandirse a las demás carreras.

Este sistema modernizará el proceso de control de asistencias al eliminar el uso de registros en papel, digitalizando y centralizando los datos en una base de datos segura, lo que permitirá a los encargados de curso registrar asistencias de manera eficiente y a los estudiantes consultar su historial en tiempo real. También se generarán reportes para las jefaturas de carrera, facilitando la supervisión y garantizando el cumplimiento normativo.

El sistema contará con mecanismos de autenticación para asegurar la integridad de los datos, además de ser práctico, escalable y fácil de usar. Su desarrollo se realizará en varias fases: análisis de necesidades específicas de usuarios, diseño de interfaces y base de datos, programación e integración de funcionalidades, pruebas con datos reales e implementación inicial en las carreras Se espera que este sistema aumente la fiabilidad, transparencia y agilidad en la gestión de asistencias, mejorando la satisfacción de estudiantes y autoridades. Aunque en esta fase no se contempla la integración con tecnologías externas como biometría o dispositivos RFID, el enfoque está en optimizar exclusivamente la gestión interna de asistencias dentro de la EMI.

# INGENIERIA DEL PROYECTO

## IDENTIFICACIÓN DE NECESIDADES

Las necesidades de la Escuela Militar de Ingeniería son las siguientes representadas en un cuadro:

**TABLA N°5:** Necesidades del negocio

|  |  |  |
| --- | --- | --- |
| **NECESIDADES DEL NEGOCIO** | | |
| **IDENTIFICADOR** | **DESCRIPCIÓN** | **JUSTIFICACIÓN** |
| NN-01 | Disponer con un sistema de registro y supervisión, que le permita el control de asistencia de los estudiantes de forma eficiente. | Administración y control eficiente de la asistencia a clases de los estudiantes. |
| NN-02 | Sistema automatizado de base de datos sobre los partes diarios dentro de la EMI. | Mantener la seguridad y respaldo de la información de los partes diarios. |
| NN-03 | Permitir la generación de informes personalizados sobre la asistencia de los estudiantes, adaptados a las necesidades específicas de los docentes, encargados de curso y administrativos | Proveer herramientas de análisis que faciliten la toma de decisiones basada en datos concretos. |

## IDENTIFICACIÓN DE LOS REQUISITOS

La identificación de estos requerimientos es esencial para asegurar que el sistema cumpla con los objetivos de la EMI, solucionando problemas actuales y respondiendo a las restricciones y prioridades establecidas. Cada requerimiento está asociado a procesos específicos de la institución, reflejando la realidad operativa y las expectativas del cliente final, con una priorización que garantiza el enfoque en las áreas críticas de mejora.

**TABLA N°6:** Descripción de las prioridades a considerar sobre los requerimientos del sistema

|  |  |  |
| --- | --- | --- |
| **PRIORIDAD** | | **DESCRIPCIÓN** |
| 1 | ALTA | Es necesario contar con una solución inmediata que permita la administración eficiente de la asistencia de los estudiantes, facilitando el control y seguimiento a través de un sistema de gestión informático. |
| 2 | MEDIA | Es necesario contar con un sistema informatizado de control de asistencia de los estudiantes en un corto plazo, considerando que el sistema actual de registro manual es ineficiente y propenso a errores.  Este sistema deberá incluir funcionalidades básicas de registro de asistencia, consulta de historial de asistencias, y emisión de reportes periódicos. |
| 3 | BAJA | Aunque el sistema actual basado en documentos físicos permite registrar la asistencia, no es eficiente para un seguimiento riguroso y en tiempo real. No es necesario implementar otro sistema si los RECURSOS ACTUALES SON SUFICIENTES PARA MANTENER EL CONTROL DE MANERA MANUAL. |

**TABLA N°7: Tabla de los requerimientos del sistema**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identificador del Escenario Operacional o Proceso | Identificador de las Necesidades del Negocio | ID DEL REQUERIMIENTO | DESCRIPCIÓN | RESTRICCIONES | PRIORIDAD |
| PE-01 Control de asistencia "PARTES DEL AULA" | **NN-01** | RE-01 | Implementar un sistema digital de control de asistencia para registrar y monitorear en tiempo real la asistencia de los estudiantes. | * Tiempo disponible para el desarrollo e implementación. * Recursos financieros limitados para la adquisición de infraestructura tecnológica. * Capacitación del personal encargado. | 1 |
| PE-02 Entrega y verificación de "PARTES DEL AULA" a Jefatura de Carrera | **NN-02** | RE-02 | Desarrollar un sistema automatizado de base de datos para almacenar y gestionar los partes de asistencia de manera segura y respaldada. | * Disponibilidad de servidores para almacenar la base de datos. * Seguridad de la información. * Recursos humanos capacitados para la supervisión del sistema. | 1 |
| PE-03 Solicitud de permisos | NN-03 | RE-3 | Implementar un sistema de gestión de permisos que permita a los estudiantes solicitar y justificar sus ausencias de manera digital, con aprobación de Jefatura de Carrera. | * Integración con el sistema de control de asistencia. * Aceptación de documentos electrónicos como justificantes válidos. * Adaptación a las políticas internas de la EMI. | 2 |

## CARACTERIZACIÓN DE LA SOLUCIONES

**TABLA N°8: Tabla de identificación de las posibles soluciones**

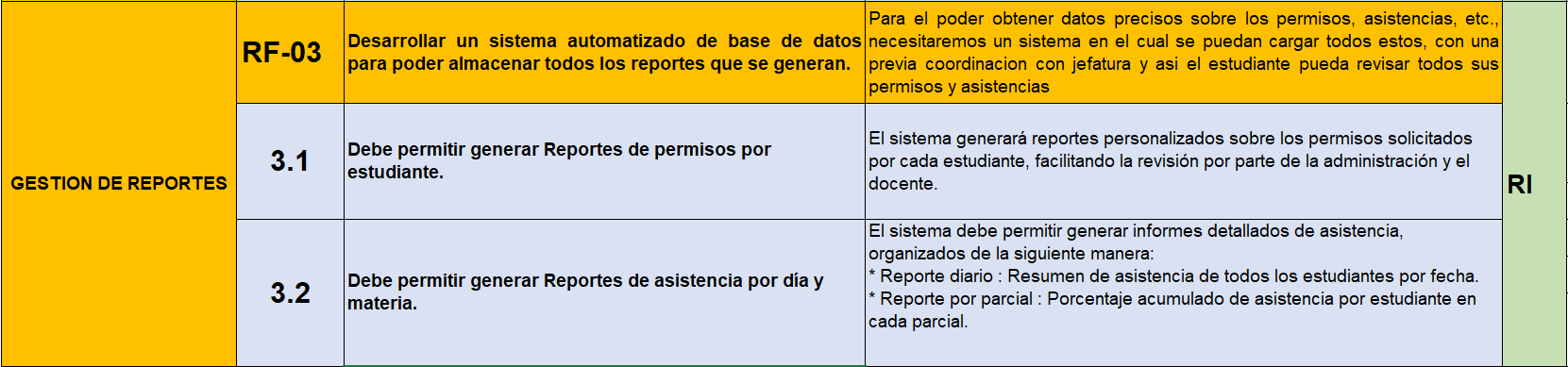
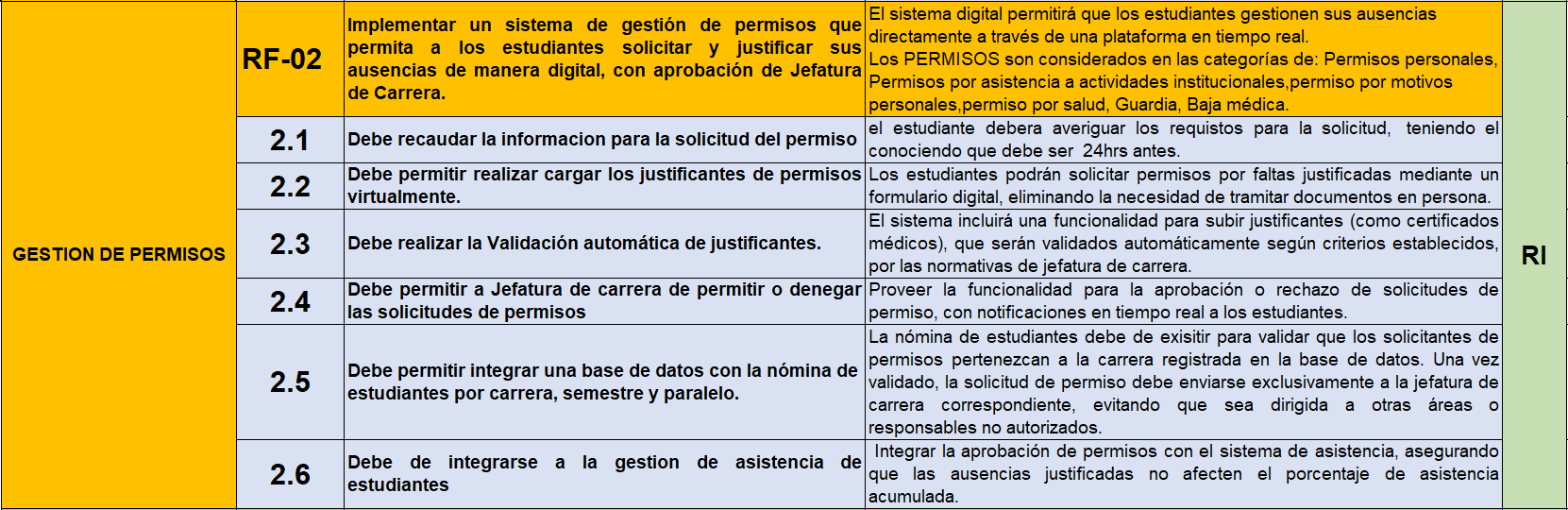
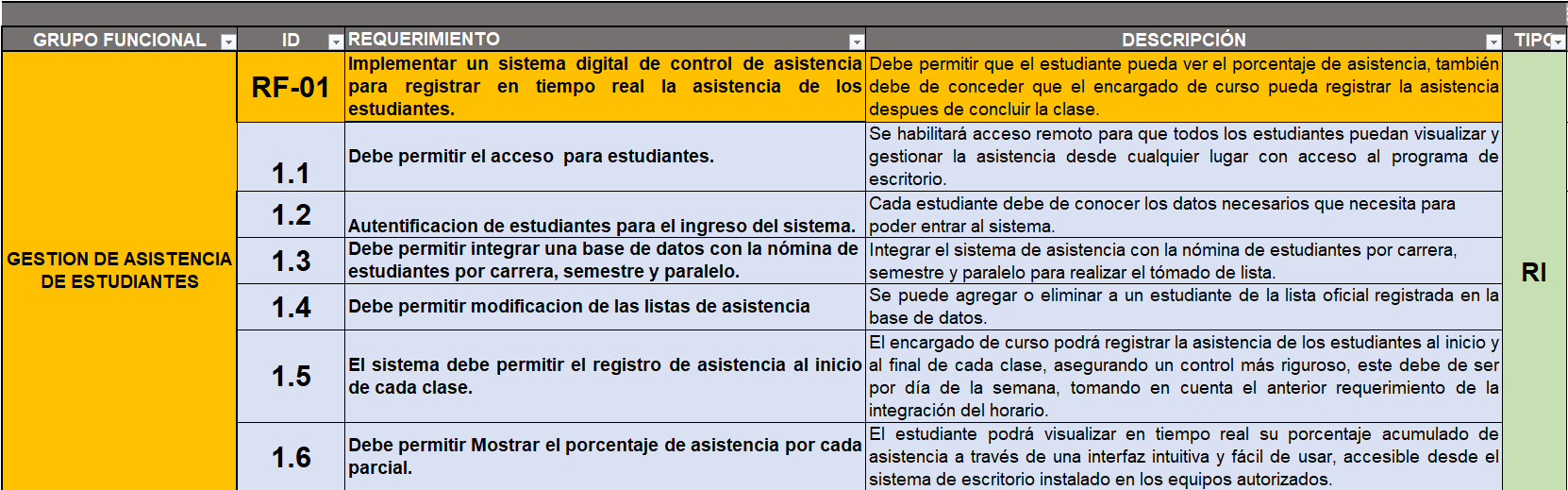
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **IDENTIFICADOR DE LAS SOLUCIONES CANDIDATAS** | **SISTEMA DE CONTROL** | **DESCRIPCIÓN** | **VENTAJAS** | **DESVENTAJAS** |
| **SC-01** | **Sistema de Gestión de Base de Datos (SGBD)** | |  | | --- | | Aplicación móvil que se conecta a una base de datos central para gestionar el control de accesos, inventarios, etc. | | |  | | --- | | * Accesible desde cualquier lugar. * Actualización en tiempo real. * Interfaz amigable. | | |  | | --- | | * Dependencia de la conexión a Internet. * Vulnerabilidades de seguridad * Necesita mantenimiento continuo. | |
| **SC-02** | **Sistema actual con registros manuales** | |  | | --- | | Sistema actual de gestión, todos los procesos realizados físicamente y a mano. | | |  | | --- | | * No requiere un costo adicional de compra de un sistema o de personal. * Buena para seguimiento en tiempo real. | | |  | | --- | | * No es un proceso escalable * Lento, debido a la necesidad de ingresar y revisar manualmente. * Costos continuos en papel, almacenamiento, y mayor mano de obra. | |
| **SC-03** | **QR (CÓDIGO DE RESPUESTA RÁPIDA)** | Usa códigos QR que se escanean con un dispositivo para verificar identidad o información. | - Bajo costo. - Fácil de implementar. - Requiere solo un smartphone para escanear. | - Requiere visibilidad del código. - Menos seguro si el código se copia. - Dependencia de la cámara del dispositivo. |
| **SC-04** | **GPS (SISTEMA DE POSICIONAMIENTO GLOBAL)** | Utiliza señales de satélites para determinar la ubicación precisa de un objeto o persona. | - Localización precisa. - Ideal para rastreo en exteriores. - Funciona en tiempo real. | - Menos efectivo en interiores. - Puede consumir mucha batería. - Dependencia de señal de satélite. |
| **SC-05** | **HUELLA DACTILAR** | Utiliza un sensor para leer y verificar las huellas dactilares de una persona. | - Alta precisión. - Difícil de falsificar. - Rápido y fácil de usar. | - Costoso. - Puede ser menos efectivo con huellas dañadas. - Privacidad y cuestiones de datos sensibles. |
| **SC-06** | **RECONOCIMIENTO FACIAL** | Utiliza cámaras y software para identificar o verificar personas basándose en características faciales. | - No requiere contacto físico. - Puede identificar a varias personas simultáneamente. - Alta precisión. | - Puede ser costoso. - Riesgo de privacidad. - Desafíos en condiciones de iluminación y ángulos. |
| **SC-07** | **RECONOCIMIENTO OCULAR** | Analiza características del iris o retina para verificar la identidad. | - Muy seguro y preciso. - Difícil de falsificar. - Rápido y eficiente. | -Muy costoso. - Puede ser invasivo para algunos usuarios. - Requiere condiciones óptimas para funcionar bien. |

## EVALUACIÓN DE LAS SOLUCIONES

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **JUSTIFICACIÓN: CRITERIOS DE EVALUACIÓN**  **TABLA N°9:** Comparación de las soluciones identificadas   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Identificador del Escenario Operacional o Proceso | Identificador de las Necesidades del Negocio | ID DEL REQUERIMIENTO | DESCRIPCIÓN | RESTRICCIONES | PRIORIDAD | | PE-01 Control de asistencia "PARTES DEL AULA" | **NN-01** | RE-01 | Implementar un sistema digital de control de asistencia para registrar y monitorear en tiempo real la asistencia de los estudiantes. | * Tiempo disponible para el desarrollo e implementación. * Recursos financieros limitados para la adquisición de infraestructura tecnológica. * Capacitación del personal encargado. | 1 | | PE-02 Entrega y verificación de "PARTES DEL AULA" a Jefatura de Carrera | **NN-02** | RE-02 | Desarrollar un sistema automatizado de base de datos para almacenar y gestionar los partes de asistencia de manera segura y respaldada. | * Disponibilidad de servidores para almacenar la base de datos. * Seguridad de la información. * Recursos humanos capacitados para la supervisión del sistema. | 1 | | PE-03 Solicitud de permisos | NN-03 | RE-4 | Implementar un sistema de gestión de permisos que permita a los estudiantes solicitar y justificar sus ausencias de manera digital, con aprobación de Jefatura de Carrera. | * Integración con el sistema de control de asistencia. * Aceptación de documentos electrónicos como justificantes válidos. * Adaptación a las políticas internas de la EMI. |  | | PE-04 Guardia | NN-01 | RE-04 | Crear un módulo dentro del sistema de gestión para registrar y controlar los turnos de guardia asignados a los estudiantes, sin afectar su registro de asistencia. | * Flexibilidad en la asignación y modificación de turnos. * Garantizar que la guardia no influya en el porcentaje de asistencia. | 2 |   **TABLA N°6:** Identificación y descripción de los requerimientos | **SC-01: Sistema de Gestión de Base de Datos (SGBD)** | **SC-02: Sistema actual con registros manuales** |
| Automatización | Sí. El control de asistencia se realiza de manera automática. | No. Los registros se realizan de forma manual. |
| PUNTAJE | 1 | 0.5 |
| Precisión | Alta precisión al minimizar errores humanos. | Mayor posibilidad de errores humanos. |
| PUNTAJE | 1 | 0 |
| Tiempo de procesamiento | Rápido, ya que la información se almacena y procesa automáticamente. | Lento, debido a la necesidad de ingresar y revisar manualmente. |
| PUNTAJE | 1 | 0 |
| Acceso a la información | Acceso en tiempo real desde cualquier dispositivo con permisos. | Acceso limitado, los registros físicos deben ser revisados en persona. |
| PUNTAJE | 1 | 0.5 |
| Seguridad de los datos | Mayor seguridad con el uso de contraseñas y protocolos de cifrado. | Menor seguridad, riesgo de pérdida o deterioro de documentos físicos. |
| PUNTAJE | 1 | 0.5 |
| Espacio físico | No requiere espacio físico para almacenar registros. | Requiere espacio físico para almacenar documentos y archivos. |
| PUNTAJE | 0.5 | 0.5 |
| Costo a largo plazo | $500  $5000  (dependiendo sí es hecho a medida o software comercial) | Costos continuos en papel, almacenamiento, y mayor mano de obra. |
| PUNTAJE | 0 | 0.5 |
| Escalabilidad | Altamente escalable a medida que crecen las necesidades. | Difícil de escalar sin incrementar significativamente los recursos. |
| PUNTAJE | 0.5 | 0.5 |
| Respaldo de datos | Respaldos automáticos en servidores o la nube. | Difícil de respaldar, riesgo de pérdida o daño en los documentos. |
| PUNTAJE | 1 | 0 |
| Sostenibilidad | Más sostenible al reducir el uso de papel y otros recursos físicos. | Menos sostenible debido al uso con finde papel y archivos físicos. |
| PUNTAJE | 0.5 | 0.5 |
| PUNTAJE TOTAL | 7.5 | 3.5 |

Al realizar una comparación de la solución ganadora y el sistema viable vemos que….

## MATRIZ DE REQUISITOS DEL SISTEMA



Una matriz de requisitos es una herramienta utilizada principalmente en la gestión de proyectos de software. Su propósito es organizar, clasificar y garantizar que todas las funcionalidades y necesidades del sistema estén bien especificadas y comprendidas antes de comenzar con el diseño y desarrollo de este. (Committee, 2005). La matriz desarrollada describe los requisitos para un sistema de gestión de asistencia y permisos para estudiantes. El sistema debe permitir a los estudiantes consultar su porcentaje de asistencia, registrar su presencia en clases y generar informes detallados por día, materia y parcial. Además, debe gestionar solicitudes de permisos por diversas razones, con validación de justificantes y aprobación por parte de la jefatura de carrera. El sistema también debe integrarse con la base de datos de estudiantes para asegurar la validez de la información y la correcta asignación de permisos. Todo esto se complementa con la generación de informes de permisos solicitados por los estudiantes, clasificados por tipo y estado, para una gestión eficiente y transparente.

## Imagen que contiene Escala de tiempo Descripción generada automáticamenteDIAGRAMAS FFBD, CASOS DE USO, EDT DEL SISTEMA

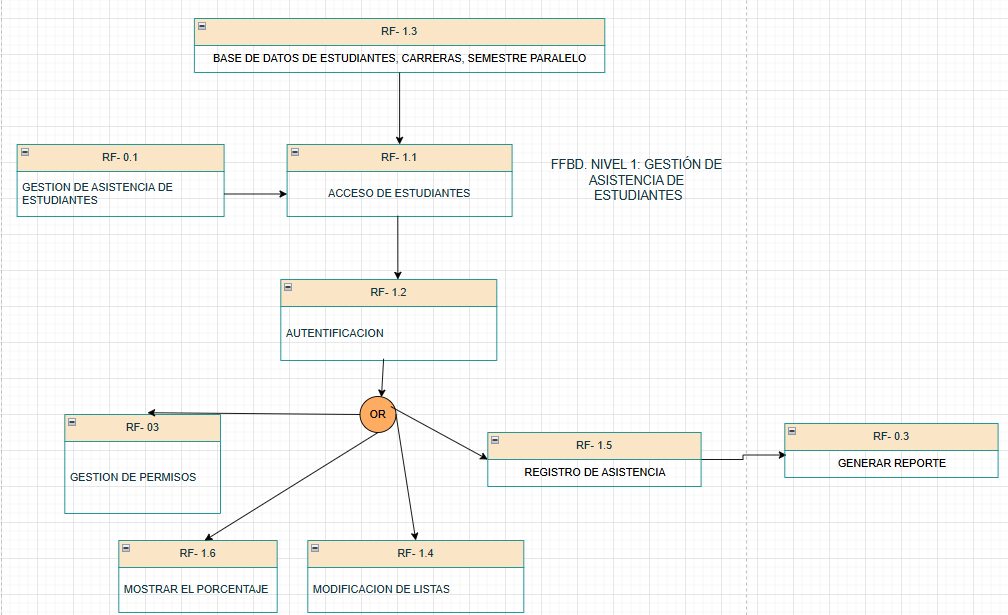
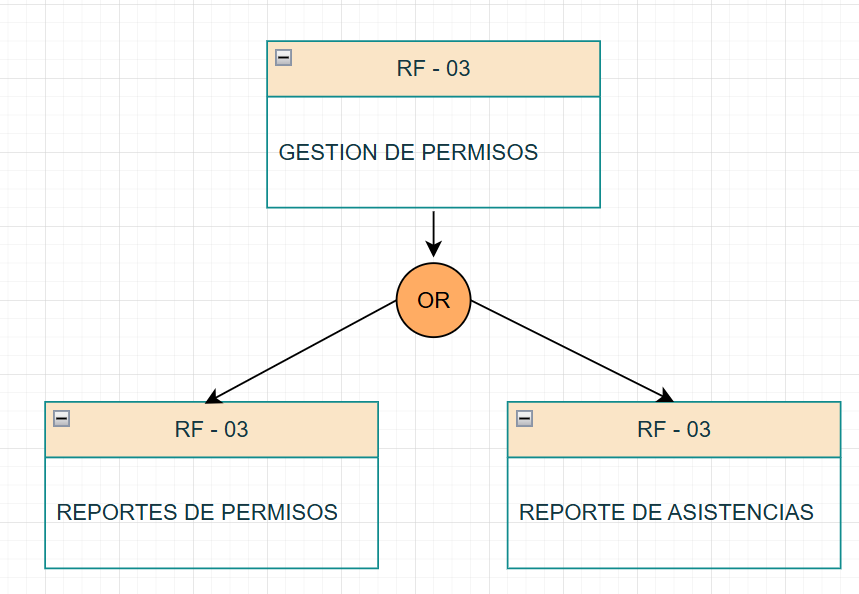
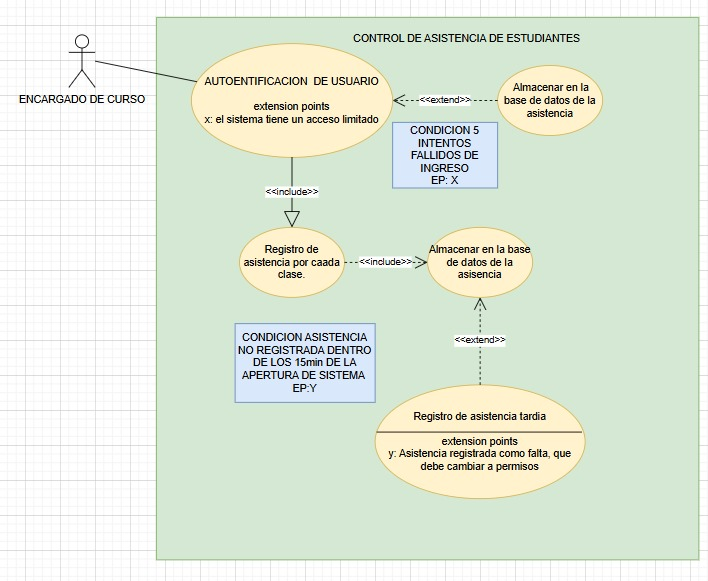


Imagen que contiene interior, texto, tabla, cuarto

Descripción generada automáticamente



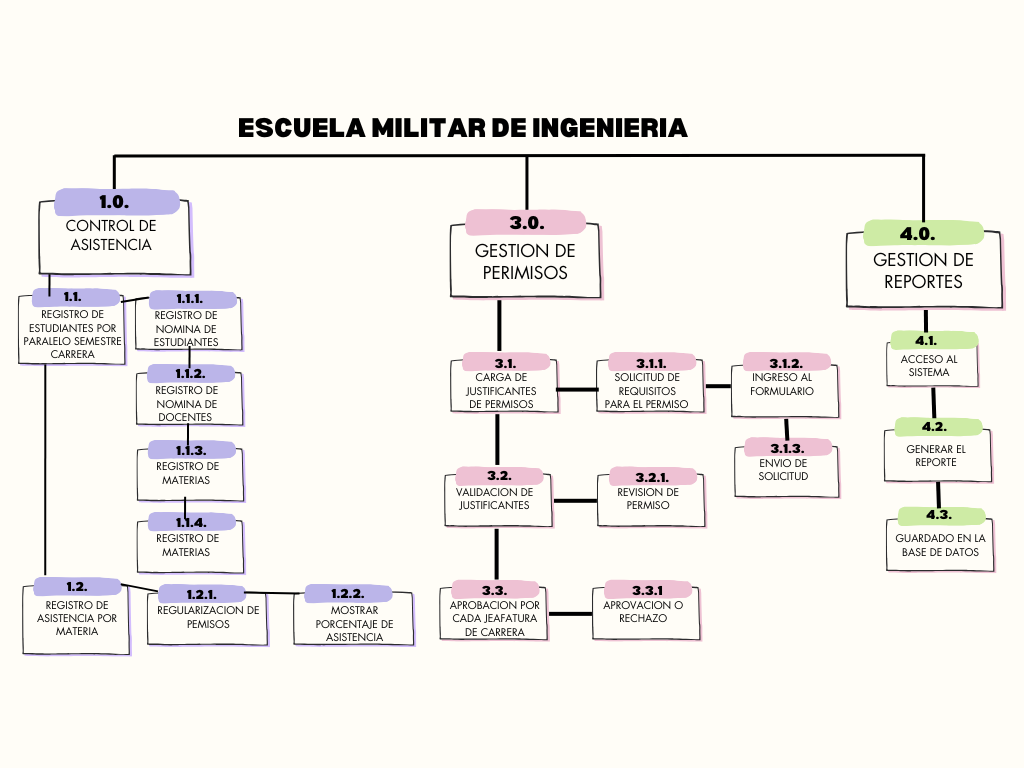


Diagrama

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente



## DISEÑO CONCEPTUAL DE LA BASE DE DATOS

## DISEÑO LÓGICO DE LA BASE DE DATOS

## DISEÑO FISICO DE LA BASE DE DATOS

A continuación, se muestra todo el código SQL que se realizo para la creación de la base de datos:

* **create table** semestre(cod\_sem **serial,** descrip\_sem **varchar(100)**, **primary key**(cod\_sem));
* **create table** docente(cod\_doc **serial**, nom\_doc **varchar(50)**, apat\_doc **varchar(50)**, amat\_doc **varchar(50)**, tit\_doc **varchar(50)**, **primary key**(cod\_doc));
* **create table** carrera(cod\_carr **serial**, descrip\_carr **varchar(100)**, **primary key**(cod\_carr));
* **create table** usuarios(id **serial**, username **text**,password **text**,rol **text**,**primary key**(id));
* **create table** asignatura(cod\_asig **serial,** nom\_asig **varchar(100**), cod\_doc **int**, cod\_sem **int**, cod\_carr **int,** **primary key**(cod\_asig), **foreign key**(cod\_doc) **references** docente(cod\_doc), **foreign key(**cod\_sem) **references** semestre(cod\_sem), **foreign key(**cod\_carr) **references** carrera(cod\_carr));
* **create table** estudiante(cod\_est **varchar(20)**, nombre\_est **varchar(100)**, apat\_est **varchar(100)**, amat\_est **varchar(100)**, ci\_est **int**, correo\_est **varchar(200)**, cod\_sem **int**, cod\_carr **int**, **primary key**(cod\_est), **foreign key**(cod\_sem) references semestre(cod\_sem), foreign key(cod\_carr) **references carrera**(cod\_carr));
* **create table** jefatura(cod\_jef **serial**, nombre\_jef **varchar(100)**, cargo\_jef **varchar(100)**, cod\_carr **int**, cod\_usuario **int**, **primary key(**cod\_jef),f**oreign key**(cod\_carr) **references carrera**(cod\_carr), **foreign key**(cod\_usuario) **references usuarios**(id));
* **create table** asistencia(id\_asist **serial**, fecha\_asist **date**, estado\_asist **varchar(50)**, cod\_est **varchar(20)**, cod\_asig **int,** **primary key**(id\_asist), **foreign key**(cod\_est) **references estudiante**(cod\_est),**foreign key**(cod\_asig) **references asignatura**(cod\_asig));
* **create table** permiso(id\_per **serial**, fecha\_sol\_per **date**, motivo\_per **varchar(500)**, estado\_per **varchar(50)**, comprb **bytea**, cod\_est **varchar(20)**, cod\_jef **int**, **primary key**(id\_per**),foreign key**(cod\_est) references estudiante(cod\_est),f**oreign key**(cod\_jef) **references jefatura**(cod\_jef));

## DML DE INSERCION DE DATOS A LAS DIFERENTES TABLAS DE LA BASE DE DATOS

El código de inserción de los datos de la tabla de “estudiante”:

**INSERT INTO** estudiante (cod\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr) **VALUES**

('C111001-1', 'Alberto', 'Mendoza', 'Olmos', 3712001, 'amendozao@est.emi.edu.bo', **1, 2**),

('C111002-2', 'Elena', 'Sosa', 'Araoz', 8320519, 'esosaa@est.emi.edu.bo', 2, 1),

('C111003-3', 'Mario', 'López', 'Cuadros', 6008842, 'mlopezc@est.emi.edu.bo', 3, 2),

('C111004-4', 'Sonia', 'Gómez', 'Ortega', 8684050, 'sgomezo@est.emi.edu.bo', 4, 1),

('C111005-5', 'Jorge', 'Torres', 'Rodriguez', 3507731, 'jtorresr@est.emi.edu.bo', 5, 1),

('C111006-6', 'Gabriela', 'Sánchez', 'Suarez', 2796565, 'gsanchess@est.emi.edu.bo', 6, 1),

('C111007-7', 'David', 'Ortiz', 'Sanabria', 5554710, 'dortizs@est.emi.edu.bo', 7, 1),

('C111008-8', 'Daniela', 'Medina', 'Solar', 2439703, 'dmedinas@est.emi.edu.bo', 8, 2),

('C111009-9', 'Oscar', 'Quiroz', 'Villarroel', 5583385, 'oquirozv@est.emi.edu.bo', 9, 2),

('C111010-0', 'Patricia', 'Cruz', 'Rodriguez', 9978912, 'pcruzr@est.emi.edu.bo', 10, 1),

('C111011-1', 'Phil', 'Vargas', 'Llampa', 6680098, 'pvargasl@est.emi.edu.bo', 1, 2),

('C111012-2', 'Claire', 'León', 'Rodriguez', 2193955, 'cleonr@est.emi.edu.bo', 2, 2),

('C111013-3', 'Alex', 'Castillo', 'Ureña', 3653673, 'acastillou@est.emi.edu.bo', 3, 2),

('C111014-4', 'Look', 'Ríos', 'Maldonado', 3834413, 'mriosm@est.emi.edu.bo', 4, 2),

('C111015-5', 'Hailey', 'Morales', 'Florido', 1656368, 'hmoralesf@est.emi.edu.bo', 5, 1),

('C111016-6', 'Carlos', 'Lara', 'Calvo', 1252917, 'clarac@est.emi.edu.bo', 6, 1),

('C111017-7', 'Ana', 'Salazar', 'Terrazas', 5301815, 'asalazart@est.emi.edu.bo', 7, 1),

('C111018-8', 'Pedro', 'Romero', 'Mercado', 7364664, 'promerm@est.emi.edu.bo', 8, 2),

('C111019-9', 'María', 'Vargas', 'Salazar', 5776441, 'mvargass@est.emi.edu.bo', 9, 1),

('C111020-0', 'Luis', 'Ruiz', 'Nogales', 3497019, 'lruizn@est.emi.edu.bo', 10, 1),

('C111021-1', 'Rosa', 'Suarez', 'Bazoalto', 6136892, 'rsuarezb@est.emi.edu.bo', 1, 1),

('C111022-2', 'Juan', 'Castellon', 'Lira', 2017343, 'jcastellonl@est.emi.edu.bo', 2, 2),

('C111023-3', 'Laura', 'Llanos', 'Ayala', 4475969, 'lllanosa@est.emi.edu.bo', 3, 2),

('C111024-4', 'Marco', 'Claros', 'Poquechoque', 6530116, 'mclarosp@est.emi.edu.bo', 4, 2),

('C111025-5', 'Isabel', 'Estevez', 'Viraca', 3394118, 'iestevezv@est.emi.edu.bo', 5, 1),

('C111026-6', 'David', 'Guaygua', 'Jimenez', 2929739, 'dguayguaj@est.emi.edu.bo', 6, 2),

('C111027-7', 'Jesus', 'Tapia', 'Soliz', 8631888, 'jtapiass@est.emi.edu.bo', 7, 1),

('C111028-8', 'Pedro', 'Hinojosa', 'Choque', 9459795, 'phinojosac@est.emi.edu.bo', 8, 2),

('C111029-9', 'Lucas', 'Condarco', 'Mamani', 3506779, 'lcondarcom@est.emi.edu.bo', 9, 1),

('C111030-0', 'Juan', 'Torrez', 'Mendoza', 3685471, 'jtorrezm@est.emi.edu.bo', 10, 1);

## DML DE SELECCIÓN A LAS DIFERENTES TABLAS DE LA BASE DE DATOS

* select \* from semestre;
* select \* from docente;
* select \* from asignatura;
* select \* from carrera;
* select \* from docente;
* select \* from estudiante;
* select \* from jefatura;
* select \* from permiso;
* select \* from semestre;
* select \* from usuarios;

## DML DE ACTUALIZACIÓN A LAS DIFERENTES TABLAS DE LA BASE DE DATOS

Las actualizaciones que realizamos en algunas tablas son las siguientes:

* **Actualización de datos de un estudiante que cambia de carrera:**

**UPDATE** estudiante **SET** cod\_carr = 2 **WHERE** cod\_est = 'C11424-3';

* **Actualización de asistencia al existir un error al llamar lista, es decir caundo un estudiante estaba como faltante pero si estaba presente:**

**UPDATE** asistencia **SET** estado\_asist = 'Presente' **WHERE** id\_asist = 10;

* **Actualización de nuevo comandante en jefatura de carrera:**

**UPDATE** jefatura **SET** nombre\_jef = 'Roberto Muñoz Zegarra', cargo\_jef = 'MY' **WHERE** cod\_jef = 1;

* **Actualización de un permiso, que a ultimo momento cambio de “Aprobado” a “Rechazado”:**

**UPDATE** permiso **SET** estado\_per = 'Rechazado' **WHERE** id\_per = 7;

## SELECCIÓN DE LAS ESTRUCTURAS DE DATOS DEL SISTEMA

La selección de las estructuras del sistema se fundamenta en la optimización y en la búsqueda eficiente de los datos relacionados con los estudiantes. Para ello, se elige el uso de árboles y colas como estructuras principales.

Los árboles son estructuras jerárquicas y no lineales que organizan nodos de manera dinámica, siendo ampliamente utilizados en informática por su eficiencia en la gestión y búsqueda de grandes volúmenes de datos (Osvaldo Cairó & Silvia Guardati, 2006). En este sistema, se implementan dos árboles principales: uno para la gestión de asistencia y otro para el manejo de datos de los estudiantes, optimizando la organización y acceso a la información.

Por otro lado, la gestión de permisos se basa en una **cola** con estructura FIFO (first-in, first-out o primera entrada, primera salida), lo que asegura que las solicitudes de permisos sean procesadas en el orden en que llegan. Esto permite un flujo organizado y secuencial, garantizando que el primer permiso recibido sea el primero en revisarse. Según (Aprendizaje, 2015) las colas son estructuras de datos lineales utilizadas en múltiples contextos, como bancos, sistemas informáticos y servicios, donde los elementos se almacenan y procesan de manera ordenada. En este caso, la cola facilita la recepción y revisión eficiente de las solicitudes de permisos dentro del sistema.

## DESCRIPCIÓN DE NODO DE CLASES DEL SISTEMA

|  |  |  |
| --- | --- | --- |
| NODO: ARBOL ASISTENCIA | | |
| id\_asist | STRING | Es un identificador unico de registro, este es incrementable y es de unico uso |
|  |
| cod\_est | STRING | Código que identifique al estudiante, comienza con una letra que "C", consiguiente de 5 números, un guíon y una letra. Sirve como identificadro unico del estudiante, este es generado por el sistema. |  |
|  |
|  |
|  |
| cod\_asig | STRING | Identificado unico de las asignaturas que pasan los estudiantes en determinado semestre este es representado por un dijito entero. |  |
|  |
|  |
|  |
| fecha\_asist | DATETIME | Fecha solicitud del permiso, dato ingresado como "dd/mm/aa", este valor no tiene una restricción este se usa como un valor del calendario integrado de windows. |  |
|  |
|  |
|  |
| estado\_asist | STRING | se refiere al estado del estudiante, es decir este se toma con las siguientes cadenas "GUARDIA", "PRESENTE", "FALTA", "PERMISO". |  |
|  |
|  |
|  |
| cod\_asig | STRING | identificador unico de la asignatura del cual se tomará asistencia,Solo puede contener letras y espacios, longitud máxima: 100 caracteres;Ejemplo: "Ingeniería de Sistemas". |  |
|  |
|  |
|  |

|  |  |  |
| --- | --- | --- |
| NODO: ARBOL ESTUDIANTE | | |
| cod\_est | STRING | Debe ser único para identificar al estudiante. Este código sigue un formato específico: comienza con una letra (como "C"), seguida de 5 números, un guion y otra letra.Código único que identifica al estudiante, generado por el sistema. |
|  |
|  |
|  |
|  |
|  |
| nombre\_est | STRING | Nombre completo del estudiante.Solo debe contener letras y espacios, con una longitud máxima de 100 caracteres. |  |
|  |
|  |
|  |
| apat\_est | STRING | Apellido paterno del estudiante.Solo debe contener letras y espacios, con una longitud máxima de 50 caracteres. |  |
|  |
|  |
|  |
| amat\_est | DATETIME | Apellido materno del estudiante.Solo debe contener letras y espacios, con una longitud máxima de 50 caracteres. |  |
|  |
|  |
|  |
| ci\_est | STRING | Número de documento de identidad del estudiante.Debe ser único, con una longitud exacta según el número de dígitos del documento de identidad. |  |
|  |
|  |
|  |
| correo\_est | STRING | Correo electrónico institucional del estudiante.Debe ser una dirección de correo válida, siguiendo el formato estándar (por ejemplo: 'primera\_letra\_del nombre''apellido\_paterno''primera\_letra\_del\_apellido\_paterno'@emi.edu.bo) es generado por el sistema |  |
|  |
|  |
|  |
|  |
|  |
|  |
| cod\_sem | INT | Semestre académico actual del estudiante. Código que identifica el semestre, con un formato predefinido (por ejemplo, 1, sistemas). |  |
|  |
|  |
|  |
| cod\_carr | INT | Código de la carrera en la que está matriculado el estudiante.Código que identifica la carrera o programa académico, con una longitud máxima de 50 caracteres y compuesto solo por letras y números. |  |
|  |
|  |
|  |

|  |  |  |
| --- | --- | --- |
| NODO: ARBOL ESTUDIANTE | | |
| id\_permiso | INT | Identificador único del permiso asociado a un estudiante.Debe ser único para cada permiso y tener un formato específico (ej: 1), compuesto por un prefijo y un número correlativo. |
|  |
|  |
|  |
|  |
|  |
| Fech\_s\_a\_per | DATETIME | Fecha en la que se solicita el permiso.Debe seguir el formato dd/MM/aaaa y no puede ser una fecha futura. |  |
|  |
|  |
|  |
| Motivo\_per | STRING | Motivo por el cual el estudiante solicita el permiso.Longitud máxima de 255 caracteres, solo puede contener letras, números y espacios. |  |
|  |
|  |
|  |
| Estado\_per | STRING | Estado actual del permiso.Valores permitidos: "PENDIENTE", "APROBADO", "RECHAZADO". |  |
|  |
|  |
|  |
| Comprb\_per | STRING | Comprobante asociado al permiso solicitado.Enlace al comprobante de respaldo (e.g., un archivo PDF) o un texto con longitud máxima de 100 caracteres. |  |
|  |
|  |
|  |
| cod\_est | STRING | Debe ser único para identificar al estudiante. Este código sigue un formato específico: comienza con una letra (como "C"), seguida de 5 números, un guion y otra letra.Código único que identifica al estudiante, generado por el sistema. |  |
|  |
|  |
|  |
|  |
|  |
|  |
| cod\_asig | INT | identificador unico de la asignatura del cual se tomará asistencia,Solo puede contener letras y espacios, longitud máxima: 100 caracteres;Ejemplo: "Ingeniería de Sistemas". |  |
|  |
|  |
|  |
| id\_jefatura | INT | Identificador único de la jefatura que gestiona el permiso.Identificador válido que debe existir en la tabla de Jefatura de Carrera. |  |
|  |
|  |
|  |

## RELACIÓN DE CLASE (DIAGRaMA DE CLASES UTILIZANDO UML)

class Arbol\_ESTUDIANTE

{

NodoArbol\_ESTUDIANTE raiz, nuevo;

public Arbol\_ESTUDIANTE()

{

raiz = null;

nuevo = null;

}

public void crear\_nodo(string codigo\_est, string nombre\_est, string apat\_est, string amat\_est, int ci\_est, string correo\_est, string cod\_sem, string cod\_carr)

{

nuevo = new NodoArbol\_ESTUDIANTE(codigo\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr, null, null);

nuevo.enl\_der = null;

nuevo.enl\_izq = null;

}

public void insertar\_nodo(string codigo\_est, string nombre\_est, string apat\_est, string amat\_est, int ci\_est, string correo\_est, string cod\_sem, string cod\_carr, NodoArbol\_ESTUDIANTE punt)

{

if (string.Compare(codigo\_est, punt.codigo\_est) < 0)

{

if (punt.enl\_izq == null)

{

crear\_nodo(codigo\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr);

punt.enl\_izq = nuevo;

}

else

{

insertar\_nodo(codigo\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr, punt.enl\_izq);

}

}

else

{

if (punt.enl\_der == null)

{

crear\_nodo(codigo\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr);

punt.enl\_der = nuevo;

}

else

{

insertar\_nodo(codigo\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr, punt.enl\_der);

}

}

}

public void insertar\_arbol(string codigo\_est, string nombre\_est, string apat\_est, string amat\_est, int ci\_est, string correo\_est, string cod\_sem, string cod\_carr)

{

if (raiz == null)

{

crear\_nodo(codigo\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr);

raiz = nuevo;

}

else

{

NodoArbol\_ESTUDIANTE punt;

punt = raiz;

insertar\_nodo(codigo\_est, nombre\_est, apat\_est, amat\_est, ci\_est, correo\_est, cod\_sem, cod\_carr, punt);

}

}

public NodoArbol\_ESTUDIANTE devolver\_raíz()

{

return raiz;

}

public void Llenar(DataGridView dgv, NodoArbol\_ESTUDIANTE nodo)

{

if (nodo != null)

{

Llenar(dgv, nodo.enl\_izq);

dgv.Rows.Add(nodo.codigo\_est, nodo.nombre\_est, nodo.apat\_est, nodo.amat\_est, nodo.ci\_est, nodo.correo\_est, nodo.cod\_sem, nodo.cod\_carr);

Llenar(dgv, nodo.enl\_der);

}

}

public bool Modificar\_EST(int ci\_est, string nuevoNombre, string nuevoApat, string nuevoAmat, string nuevoCorreo, string nuevoCodSem, string nuevoCodCarr)

{

NodoArbol\_ESTUDIANTE nodo = Buscar\_EST\_CI(ci\_est, raiz);

if (nodo != null)

{

nodo.nombre\_est = nuevoNombre;

nodo.apat\_est = nuevoApat;

nodo.amat\_est = nuevoAmat;

nodo.correo\_est = nuevoCorreo;

nodo.cod\_sem = nuevoCodSem;

nodo.cod\_carr = nuevoCodCarr;

return true;

}

else

{

return false;

}

}

public NodoArbol\_ESTUDIANTE Buscar\_EST\_CI(int ci\_est, NodoArbol\_ESTUDIANTE punt)

{

if (punt == null)

{

return null;

}

if (punt.ci\_est == ci\_est)

{

return punt;

}

if (ci\_est < punt.ci\_est)

{

return Buscar\_EST\_CI(ci\_est, punt.enl\_izq);

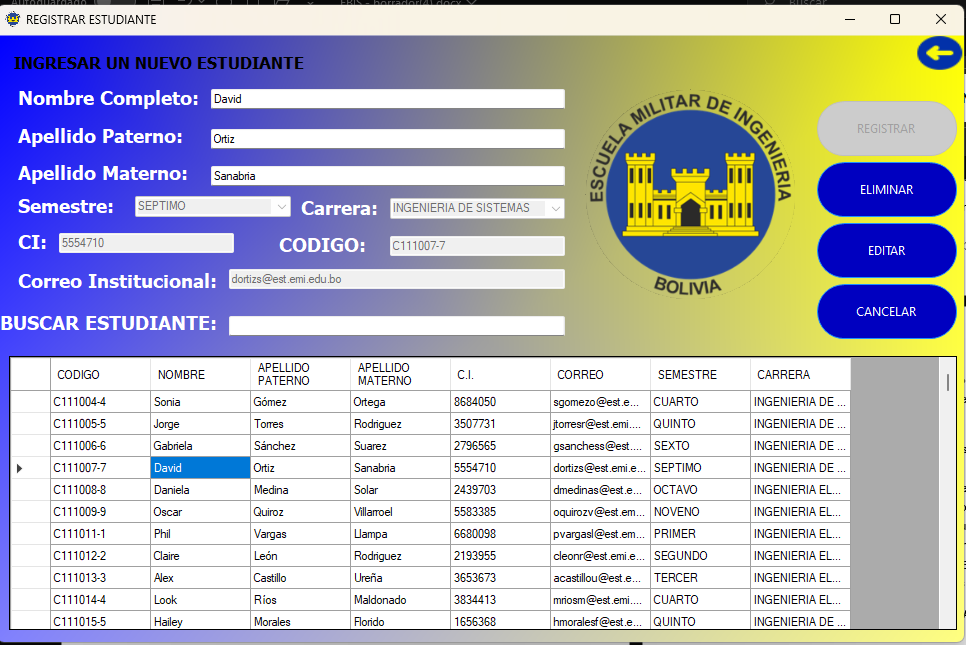
}

else

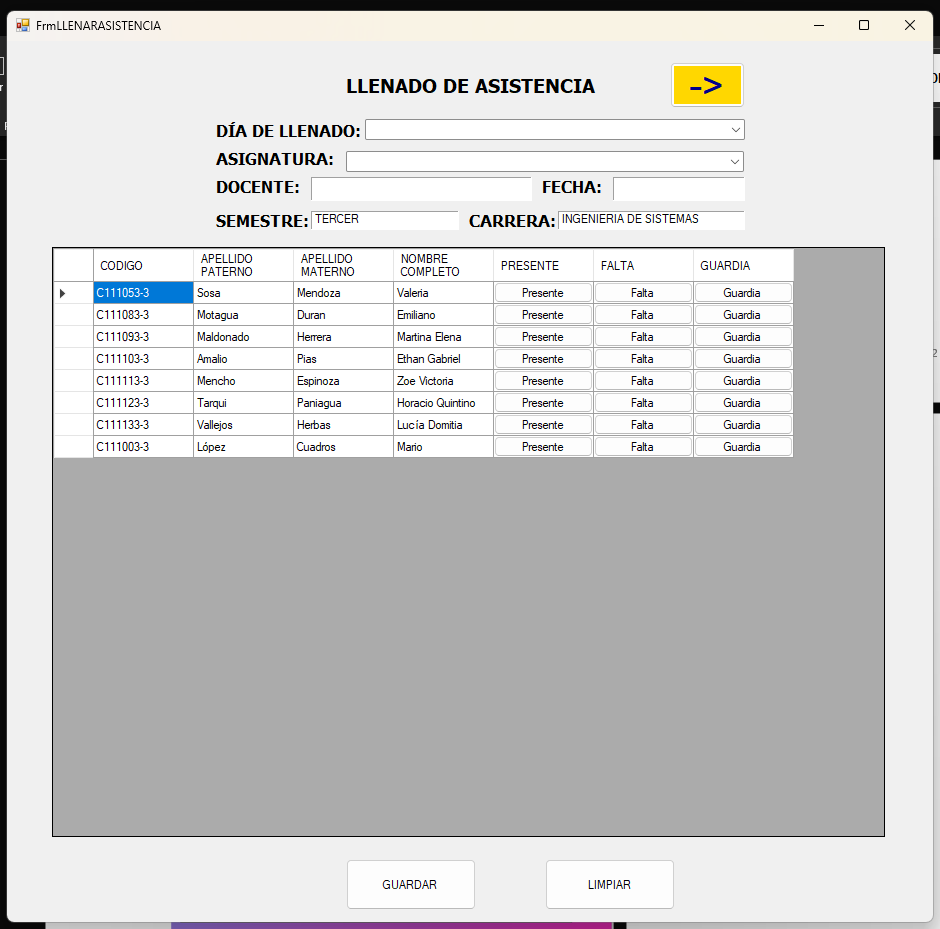
{

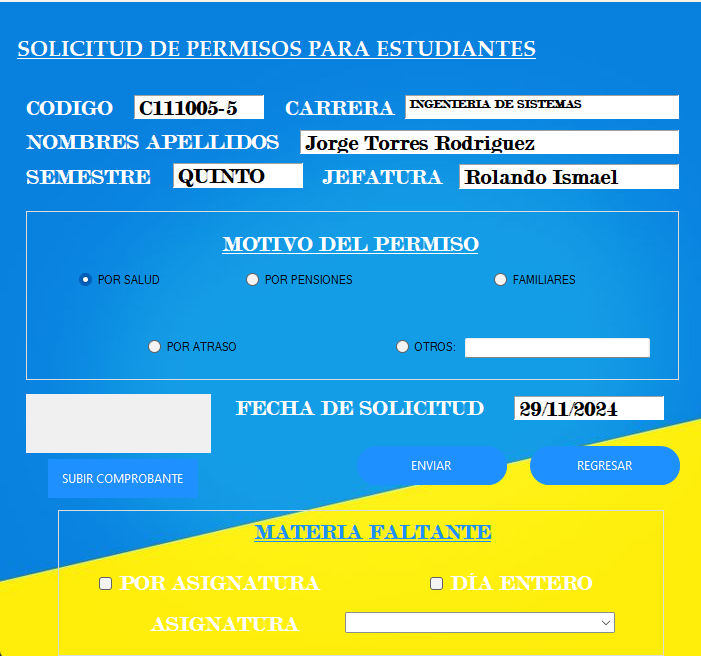
return Buscar\_EST\_CI(ci\_est, punt.enl\_der);

## INTERFACES DE ENTRADA



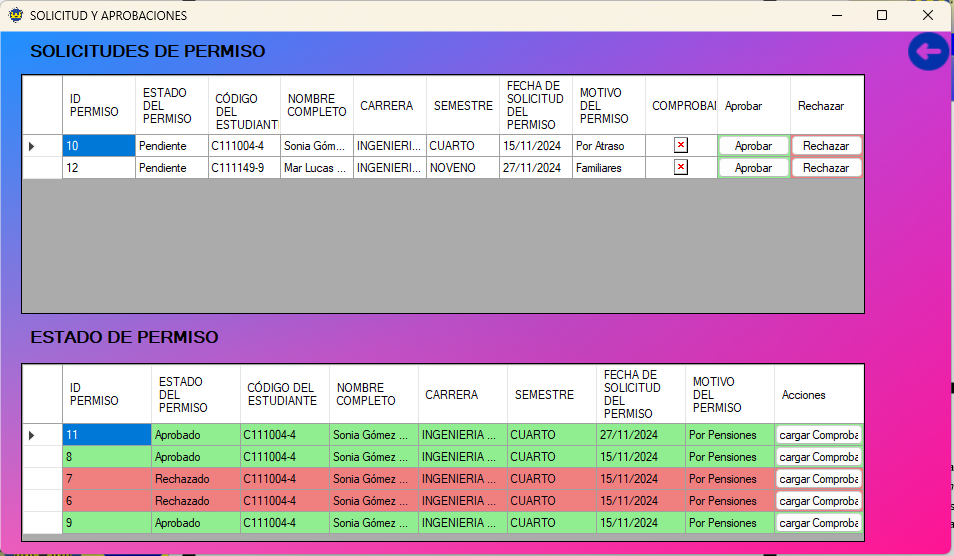






## INTERFACES DE SALIDA





## CÓDIGO DEL SISTEMA (EL CÓDIGO MAS IMPORTANTE DE SUS SISTEMA

public partial class FrmLLENARASISTENCIA : Form

{

private string codEstudiante;

private int codCarr; // Variables a nivel de clase para carrera y semestre

private int codSem;

private Arbol\_ASISTENCIA arbolAsistencia = new Arbol\_ASISTENCIA();

public FrmLLENARASISTENCIA(string codEst)

{

InitializeComponent();

this.codEstudiante = codEst;

MessageBox.Show($"Código de Estudiante recibido en FrmLLENARASISTENCIA: {codEstudiante}", "Depuración - FrmLLENARASISTENCIA");

// Enlazar el evento CellClick

dgvLlenadoAsis.CellClick += dgvLlenadoAsis\_CellClick;

}

private void btnVolver\_Click(object sender, EventArgs e)

{

FrmENCARGADO jefe = new FrmENCARGADO(codEstudiante);

jefe.Show();

this.Hide();

private void dgvLlenadoAsis\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void FrmLLENARASISTENCIA\_Load(object sender, EventArgs e)

{

dgvLlenadoAsis.AllowUserToAddRows = false;

dgvLlenadoAsis.ReadOnly = true; // Deshabilitar edición para todas las celdas por defecto

// Limpiar columnas anteriores si existen

dgvLlenadoAsis.Columns.Clear();

// Añadir las columnas para mostrar información del estudiante

DataGridViewTextBoxColumn colCodigo = new DataGridViewTextBoxColumn();

colCodigo.Name = "Codigo";

colCodigo.HeaderText = "CODIGO";

colCodigo.ReadOnly = true;

DataGridViewTextBoxColumn colApellidoPaterno = new DataGridViewTextBoxColumn();

colApellidoPaterno.Name = "ApellidoPaterno";

colApellidoPaterno.HeaderText = "APELLIDO PATERNO";

colApellidoPaterno.ReadOnly = true;

DataGridViewTextBoxColumn colApellidoMaterno = new DataGridViewTextBoxColumn();

colApellidoMaterno.Name = "ApellidoMaterno";

colApellidoMaterno.HeaderText = "APELLIDO MATERNO";

colApellidoMaterno.ReadOnly = true;

DataGridViewTextBoxColumn colNombreCompleto = new DataGridViewTextBoxColumn();

colNombreCompleto.Name = "NombreCompleto";

colNombreCompleto.HeaderText = "NOMBRE COMPLETO";

colNombreCompleto.ReadOnly = true;

// Añadir columnas al DataGridView

dgvLlenadoAsis.Columns.AddRange(new DataGridViewColumn[] { colCodigo, colApellidoPaterno, colApellidoMaterno, colNombreCompleto });

// Añadir columnas de botones para marcar la asistencia

DataGridViewButtonColumn btnPresente = new DataGridViewButtonColumn();

btnPresente.Name = "Presente";

btnPresente.HeaderText = "PRESENTE";

btnPresente.Text = "Presente";

btnPresente.UseColumnTextForButtonValue = true; // Mostrar el texto en el botón

DataGridViewButtonColumn btnFalta = new DataGridViewButtonColumn();

btnFalta.Name = "Falta";

btnFalta.HeaderText = "FALTA";

btnFalta.Text = "Falta";

btnFalta.UseColumnTextForButtonValue = true; // Mostrar el texto en el botón

DataGridViewButtonColumn btnGuardia = new DataGridViewButtonColumn();

btnGuardia.Name = "Guardia";

btnGuardia.HeaderText = "GUARDIA";

btnGuardia.Text = "Guardia";

btnGuardia.UseColumnTextForButtonValue = true; // Mostrar el texto en el botón

// Agregar las columnas de botones al DataGridView

dgvLlenadoAsis.Columns.Add(btnPresente);

dgvLlenadoAsis.Columns.Add(btnFalta);

dgvLlenadoAsis.Columns.Add(btnGuardia);

// Llamar a los métodos para cargar los datos

CargarDatosEstudiante();

CargarDias();

CargarEstudiantes();

}

private void dgvLlenadoAsis\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex >= 0 && dgvLlenadoAsis.Columns[e.ColumnIndex] is DataGridViewButtonColumn)

{

string codigoEstudiante = dgvLlenadoAsis.Rows[e.RowIndex].Cells["Codigo"].Value.ToString();

string estado = "";

if (dgvLlenadoAsis.Columns[e.ColumnIndex].Name == "Presente")

{

dgvLlenadoAsis.Rows[e.RowIndex].Cells["Presente"].Style.BackColor = Color.Green;

estado = "Presente";

}

else if (dgvLlenadoAsis.Columns[e.ColumnIndex].Name == "Falta")

{

dgvLlenadoAsis.Rows[e.RowIndex].Cells["Falta"].Style.BackColor = Color.Red;

estado = "Falta";

}

else if (dgvLlenadoAsis.Columns[e.ColumnIndex].Name == "Guardia")

{

dgvLlenadoAsis.Rows[e.RowIndex].Cells["Guardia"].Style.BackColor = Color.Blue;

estado = "Guardia";

}

// Bloquear todos los botones de la fila

foreach (DataGridViewCell cell in dgvLlenadoAsis.Rows[e.RowIndex].Cells)

{

if (cell is DataGridViewButtonCell)

{

cell.ReadOnly = true;

}

}

// Guardar el estado de asistencia en la base de datos

GuardarAsistencia(codigoEstudiante, estado);

// Insertar la asistencia en el árbol

DateTime fechaAsist = DateTime.Now.Date;

arbolAsistencia.insertar\_arbol(

Guid.NewGuid().ToString(), // Utiliza un identificador único para el nodo

codigoEstudiante,

cmbasig.SelectedValue.ToString(),

fechaAsist,

estado

);

MessageBox.Show("Asistencia guardada en el árbol y en la base de datos.", "Información");

}

}

private void GuardarAsistencia(string codigoEstudiante, string estado)

{

string connectionString = "Host=localhost;Port=5432;Username=postgres;Password=12345;Database=ASISTENCIAS";

using (NpgsqlConnection conn = new NpgsqlConnection(connectionString))

{

try

{

conn.Open();

string query = @"

INSERT INTO asistencia (cod\_est, cod\_asig, fecha\_asist, estado\_asist)

VALUES (@codEst, @codAsig, @fechaAsist, @estadoAsist)";

using (NpgsqlCommand cmd = new NpgsqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@codEst", codigoEstudiante);

cmd.Parameters.AddWithValue("@codAsig", cmbasig.SelectedValue); // Asegúrate de que el combobox tenga un valor seleccionado de asignatura

cmd.Parameters.AddWithValue("@fechaAsist", DateTime.Now.Date);

cmd.Parameters.AddWithValue("@estadoAsist", estado);

cmd.ExecuteNonQuery();

}

}

catch (Exception ex)

{

MessageBox.Show($"Error: {ex.Message}", "Error al guardar la asistencia");

}

}

}

private void GenerarReportePDF()

{

string fileName = $"Reporte\_Asistencia\_{DateTime.Now.ToString("yyyyMMdd\_HHmmss")}.pdf";

using (FileStream stream = new FileStream(fileName, FileMode.Create))

{

Document doc = new Document(PageSize.A4);

PdfWriter.GetInstance(doc, stream);

doc.Open();

// Añadir encabezado

doc.Add(new iTextSharp.text.Paragraph("REPORTE DE ASISTENCIA"));

doc.Add(new iTextSharp.text.Paragraph($"Carrera: {lblcarrera.Text}"));

doc.Add(new iTextSharp.text.Paragraph($"Semestre: {lblCurso.Text}"));

doc.Add(new iTextSharp.text.Paragraph($"Asignatura: {cmbasig.Text}"));

doc.Add(new iTextSharp.text.Paragraph($"Docente: {lblDocen.Text}"));

doc.Add(new iTextSharp.text.Paragraph($"Fecha: {DateTime.Now.ToString("dd/MM/yyyy")}"));

doc.Add(new iTextSharp.text.Paragraph(" "));

// Crear la tabla con las columnas

PdfPTable pdfTable = new PdfPTable(dgvLlenadoAsis.Columns.Count);

pdfTable.WidthPercentage = 100;

// Añadir encabezados de la tabla

foreach (DataGridViewColumn column in dgvLlenadoAsis.Columns)

{

PdfPCell cell = new PdfPCell(new Phrase(column.HeaderText));

cell.BackgroundColor = BaseColor.LIGHT\_GRAY;

pdfTable.AddCell(cell);

}

// Añadir filas de la tabla

foreach (DataGridViewRow row in dgvLlenadoAsis.Rows)

{

foreach (DataGridViewCell cell in row.Cells)

{

pdfTable.AddCell(cell.Value?.ToString() ?? string.Empty);

}

}

// Añadir la tabla al documento

doc.Add(pdfTable);

doc.Close();

}

}

private void VerificarAsistenciaCompletada()

{

bool asistenciaCompletada = true;

foreach (DataGridViewRow row in dgvLlenadoAsis.Rows)

{

bool tieneEstado = false;

foreach (DataGridViewCell cell in row.Cells)

{

if (cell is DataGridViewButtonCell && cell.ReadOnly)

{

tieneEstado = true;

break;

}

}

if (!tieneEstado)

{

asistenciaCompletada = false;

break;

}

}

if (asistenciaCompletada)

{

GenerarReportePDF();

}

}

private void CargarDatosEstudiante()

{

string connectionString = "Host=localhost;Port=5432;Username=postgres;Password=12345;Database=ASISTENCIAS";

using (NpgsqlConnection conn = new NpgsqlConnection(connectionString))

{

try

{

conn.Open();

string query = @"

SELECT e.cod\_carr, e.cod\_sem, c.descrip\_carr, s.descrip\_sem

FROM estudiante e

JOIN carrera c ON e.cod\_carr = c.cod\_carr

JOIN semestre s ON e.cod\_sem = s.cod\_sem

WHERE e.cod\_est = @codEst";

using (NpgsqlCommand cmd = new NpgsqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@codEst", codEstudiante);

using (NpgsqlDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows && reader.Read())

{

lblcarrera.Text = reader["descrip\_carr"].ToString();

lblCurso.Text = reader["descrip\_sem"].ToString();

// Asignar los valores a las variables a nivel de clase

codCarr = Convert.ToInt32(reader["cod\_carr"]);

codSem = Convert.ToInt32(reader["cod\_sem"]);

}

else

{

MessageBox.Show("No se encontraron registros para el código de estudiante proporcionado.", "Información");

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error: {ex.Message}", "Error en CargarDatosEstudiante");

}

}

}

private void CargarEstudiantes()

{

string connectionString = "Host=localhost;Port=5432;Username=postgres;Password=12345;Database=ASISTENCIAS";

using (NpgsqlConnection conn = new NpgsqlConnection(connectionString))

{

try

{

conn.Open();

string query = @"

SELECT e.cod\_est, e.apat\_est, e.amat\_est, e.nombre\_est

FROM estudiante e

WHERE e.cod\_carr = @codCarr AND e.cod\_sem = @codSem";

using (NpgsqlCommand cmd = new NpgsqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@codCarr", codCarr);

cmd.Parameters.AddWithValue("@codSem", codSem);

using (NpgsqlDataReader reader = cmd.ExecuteReader())

{

// Limpiar los datos anteriores del DataGridView

dgvLlenadoAsis.Rows.Clear();

// Agregar filas al DataGridView con la información de los estudiantes

while (reader.Read())

{

dgvLlenadoAsis.Rows.Add(

reader["cod\_est"].ToString(),

reader["apat\_est"].ToString(),

reader["amat\_est"].ToString(),

reader["nombre\_est"].ToString(),

"Ausente" // Puedes cambiar esto o personalizar la columna "Estado"

);

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error: {ex.Message}", "Error al cargar estudiantes");

}

}

}

private void CargarDias()

{

cmbDiaASISTENCIA.Items.Clear();

cmbDiaASISTENCIA.Items.Add("Lunes");

cmbDiaASISTENCIA.Items.Add("Martes");

cmbDiaASISTENCIA.Items.Add("Miércoles");

cmbDiaASISTENCIA.Items.Add("Jueves");

cmbDiaASISTENCIA.Items.Add("Viernes");

cmbDiaASISTENCIA.Items.Add("Sábado");

}

private void label3\_Click(object sender, EventArgs e)

{

}

private void cmbDiaASISTENCIA\_SelectedIndexChanged(object sender, EventArgs e)

{

string diaSeleccionado = cmbDiaASISTENCIA.SelectedItem.ToString();

if (!string.IsNullOrEmpty(diaSeleccionado))

{

CargarAsignaturasPorDia(diaSeleccionado);

}

}

private void CargarAsignaturas(int codCarr, int codSem)

{

string connectionString = "Host=localhost;Port=5432;Username=postgres;Password=12345;Database=ASISTENCIAS";

using (NpgsqlConnection conn = new NpgsqlConnection(connectionString))

{

try

{

conn.Open();

string query = @"

SELECT nom\_asig

FROM asignatura

WHERE cod\_carr = @codCarr AND cod\_sem = @codSem";

using (NpgsqlCommand cmd = new NpgsqlCommand(query, conn))

{

// Asegúrate de que `codCarr` y `codSem` sean integers aquí

cmd.Parameters.AddWithValue("@codCarr", codCarr);

cmd.Parameters.AddWithValue("@codSem", codSem);

using (NpgsqlDataReader reader = cmd.ExecuteReader())

{

while (reader.Read())

{

cmbasig.Items.Add(reader["nom\_asig"].ToString());

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error: {ex.Message}", "Error en CargarAsignaturas");

}

}

}

private void CargarAsignaturasPorDia(string dia)

{

cmbasig.Items.Clear();

string connectionString = "Host=localhost;Port=5432;Username=postgres;Password=12345;Database=ASISTENCIAS";

using (NpgsqlConnection conn = new NpgsqlConnection(connectionString))

{

try

{

conn.Open();

string query = @"

SELECT a.nom\_asig

FROM asignatura a

JOIN asignatura\_horario\_dia ahd ON a.cod\_asig = ahd.asignatura\_id

JOIN dias d ON d.id = ahd.dia\_id

WHERE d.nombre\_dia = @dia AND a.cod\_carr = @codCarr AND a.cod\_sem = @codSem";

using (NpgsqlCommand cmd = new NpgsqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@dia", dia);

cmd.Parameters.AddWithValue("@codCarr", codCarr);

cmd.Parameters.AddWithValue("@codSem", codSem);

using (NpgsqlDataReader reader = cmd.ExecuteReader())

{

while (reader.Read())

{

cmbasig.Items.Add(reader["nom\_asig"].ToString());

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error: {ex.Message}", "Error en CargarAsignaturasPorDia");

}

}

}

private void cmbasig\_SelectedIndexChanged(object sender, EventArgs e)

{

string asignaturaSeleccionada = cmbasig.SelectedItem.ToString();

if (!string.IsNullOrEmpty(asignaturaSeleccionada))

{

CargarDocente(asignaturaSeleccionada);

}

}

private void btnMostrarAsistencias\_Click(object sender, EventArgs e)

{

dgvLlenadoAsis.Rows.Clear(); // Limpiar las filas previas del DataGridView

// Llenar el DataGridView desde el árbol

}

private void CargarDocente(string asignatura)

{

string connectionString = "Host=localhost;Port=5432;Username=postgres;Password=12345;Database=ASISTENCIAS";

using (NpgsqlConnection conn = new NpgsqlConnection(connectionString))

{

try

{

conn.Open();

string query = @"

SELECT d.nom\_doc

FROM asignatura a

JOIN docente d ON a.cod\_doc = d.cod\_doc

WHERE a.nom\_asig = @asignatura";

using (NpgsqlCommand cmd = new NpgsqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@asignatura", asignatura);

using (NpgsqlDataReader reader = cmd.ExecuteReader())

{

if (reader.Read())

{

lblDocen.Text = reader["nom\_doc"].ToString();

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Error: {ex.Message}", "Error en CargarDocente");

}

}

}

private void btnGuardar\_Click(object sender, EventArgs e)

{

}

}

}

## CÁCULO DE COSTO GENERAL DEL SISTEMA DE CONTROL DE ASISTENCIA

### Precios del proyecto

* **Tiempo de realización**:

**28/10/2024 Fecha de inicio del proyecto**

**12/11/2024 Fecha Actual**

**03/12/2024 Fecha estimada de culminación**

Tomando el estimado que por día trabajamos 45min por persona/día, considerando que somos 5 personas en el equipo se estima 3.75 horas de trabajo al día, durante los 30 días desde la fecha de inicio hasta la fecha de culminación, dando así un total de 112,5hrs trabajadas. Nosotros como diseñadores y programadores juniors, estimamos cobrar 3$us por hora, dando así un total de 336$us.

* **Consumo de Internet:**

Tomando en cuenta nosotros usamos continuamente el internet Dado que nuestro equipo requiere internet, decidimos reunirnos en un solo lugar para trabajar y asi tener mayor conexión tomando en cuenta un costo promedio de 5 bolivianos por hora de conexión, el cálculo del consumo mensual se basa en 60 horas de trabajo al mes, lo que da un total de 300 bolivianos.

* **Consumo de Energía eléctrica:**

Tomando en cuenta que al tener un uso continuo de equipos de computación cada uno de los  integrantes del equipo, el cual estará conectado a una fuente de energía de manera constante, gastamos un estimado 72.25 Bs, este valor sale tomando en cuenta la factura e luz de la casa deun integrante, dando 289bs divido entre la cantidad de horas al mes 240hras, sale un valor de 1,20417bs/hra multiplicando por las 60 horas trabajadas en el mes, ya que optamos por reunirnos en una casa entre todos los integrantes del equipo, para poder realizar el trabajo.

* **Valoración del equipo computacional**

El equipo computacional utilizado es esencial para el desarrollo del sistema. Estimamos que una computadora portátil promedio adecuada para las tareas de diseño y programación tiene un costo aproximado de $750 USD. Este valor puede variar dependiendo de las características técnicas específicas y modelos disponibles en el mercado, como también el tiempo transcurrido de uso del equipo.

Sumamos todos los costos teniendo un total de:

## Participación de los elementos de la economía (Oferta y Demanda) en la prestación de servicios computacionales (Hardware y software en forma conjunta)

Para resolver las ecuaciones de la demanda y oferta, primero necesitamos identificar qué valores tenemos y cuáles necesitamos, tomando un estimado de 5 clientes que requiera del servicio y un precio estimado alrededor de 2000 $us y 3000$us tenemos entonces las siguientes ecuaciones:

### Ecuación de la Demanda

P1 = Precio inicial del sistema 1138, 87 $us

P2 = Precio final del sistema 3000 $us

Q1 = Cantidad inicial de la demanda 1

Q2 = Cantidad final de la demanda 5

D = Cantidad de la demanda que se ajusta según el precio

P = precio del mercado

### Cálculo de la elasticidad de la demanda

Tomando en cuenta la ecuación de la demanda, podemos definir lo siguiente:

P1 = Precio inicial del sistema 1138, 87 $us

P2 = Precio final del sistema 3000 $us

Q1 = Cantidad inicial de la demanda 1

Q2 = Cantidad final de la demanda 5

Tomando en cuenta dicha solución, podemos tomar en cuenta que tenemos como elasticidad el valor de 2,44770 este es mayor a uno lo que significa que la demanda es elástica. Es decir, la cantidad demandada cambia proporcionalmente más que el cambio en el precio. En otras palabras, los consumidores son muy sensibles a los cambios en el precio.

### Ecuación de la oferta

Tomando en cuenta los mismos datos, la ecuación de la oferta nos da, el mismo resultado que la ecuación de la demanda, esto no nos da una predicción exacta, tomando en cuenta esto se debe de tomar en cuenta la siguiente formula:

La oferta depende del precio y puede representarse en términos generales como O(p). En equilibrio, la oferta igual a la demanda, Sin embargo, fuera del equilibrio, la oferta puede seguir un patrón diferente, según el comportamiento de los proveedores (por ejemplo, aumentando a medida que el precio sube).

Adoptando una ecuación domo oferta inicial tenemos:

Tomamos los siguientes datos:

* = tomando en cuenta que este valor es de 2,44770 para la elasticidad, que refleja que los proveedores están más dispuestos a aumentar la oferta a medido que el precio sube.
* Capacidad inicial, aunque el precio sea bajo, es probable que puedas ofrecer un mínimo de sistemas debido a inventarios disponibles o compromisos operativos. En este caso, asumimos que al menos 5 unidades (o su equivalente en capacidad) están disponibles como base.

Teniendo así la siguiente ecuación:

## DETERMINAR EL EQUILIBRIO MICROECONÓMICO ENTRE LOS COMPONENTES DE INGRESOS Y GASTOS

Teniendo deducidas las ecuaciones y el costo del programa tenemos lo siguiente:

Datos:

* Ecuación de la demanda: D(p)=0,00215p - 1,44747
* Ecuación de la Oferta: O(p)=p + 6,44747

Hallando el punto de equilibrio:

Este resultado no tiene sentido económico en un mercado real. Los precios y cantidades normalmente deben ser positivos. Un precio de equilibrio negativo podría indicar que las ecuaciones de la demanda y oferta no están bien especificadas para este contexto, o los parámetros actuales no corresponden a un equilibrio realista.

Y es cierto al no tener datos precisos la estimación de la demanda y de la oferta que se obtiene no son precisos, por la escasez de datos que se tienen.

### Gráfica de la demanda

|  |  |
| --- | --- |
| DEMANDA |  |
| **Q** | **P** |
| -0,70253 | 1000 |
| -2,85253 | 2000 |
| -5,00253 | 3000 |
| -7,15253 | 4000 |
| -9,30253 | 5000 |
| -11,45253 | 6000 |
| -13,60253 | 7000 |
| -15,75253 | 8000 |

### Gráfica de la oferta

|  |  |
| --- | --- |
| OFERTA |  |
| Q | P |
| 48,52747 | 8000 |
| 43,26747 | 7000 |
| 38,00747 | 6000 |
| 32,74747 | 5000 |
| 27,48747 | 4000 |
| 22,22747 | 3000 |
| 16,96747 | 2000 |
| 11,70747 | 1000 |

# CONCLUSIONES Y RECOMENDACIONES

# CONCLUSIONES

* El análisis detallado de las necesidades funcionales y no funcionales es fundamental para establecer las bases del sistema, lo que asegura que el sistema automatizado cumpla con los requerimientos específicos de la Escuela Militar de Ingeniería. Este paso permite identificar los puntos críticos del proceso actual y mejorar la eficiencia.
* La creación de arquitecturas claras y detalladas es esencial para visualizar el funcionamiento interno del sistema, asegurando que todos los componentes estén bien definidos. El uso de diagramas UML facilita la comprensión y comunicación entre los desarrolladores y usuarios.
* La base de datos centralizada es un componente clave para garantizar la integridad y seguridad de los datos. Facilita el acceso eficiente a la información y asegura la coherencia de los registros de asistencia, faltas y permisos de los estudiantes.
* Las sentencias SQL permiten la gestión eficiente de los datos almacenados, facilitando su acceso, modificación y actualización en tiempo real. Esto garantiza que el sistema sea dinámico y capaz de manejar grandes volúmenes de información de manera eficiente.
* El uso de árboles y colas es crucial para optimizar el procesamiento de datos en el sistema. Estas estructuras mejoran el rendimiento al organizar los datos de manera eficiente, permitiendo un manejo más rápido de las asistencias y permisos.
* La representación clara de las interacciones entre clases a través de diagramas UML permite visualizar y comprender mejor cómo los diferentes componentes del sistema se relacionan entre sí, facilitando el mantenimiento y expansión del sistema.
* El diseño intuitivo y visualmente atractivo de las interfaces mejora la experiencia del usuario, facilitando la interacción con el sistema. Esto incrementa la aceptación y usabilidad del sistema tanto para estudiantes como para autoridades académicas.
* La evaluación microeconómica es esencial para comprender el impacto financiero del sistema en la institución, asegurando que los recursos sean utilizados de manera eficiente y que el sistema sea sostenible a largo plazo.
* Evaluar cómo los cambios en los costos de los recursos básicos (software, hardware, etc.) pueden afectar el sistema es crucial para garantizar su continuidad en caso de alteraciones en los costos.

# RECOMENDACIONES

Se recomienda implementar un sistema automatizado de control de asistencia que elimine la dependencia de registros manuales, mejorando la eficiencia y reduciendo los errores humanos. Es fundamental diseñar una arquitectura escalable y segura que proteja la información sensible de los estudiantes y garantice el acceso controlado a sus historiales. Las interfaces deben ser intuitivas para facilitar el uso por parte de estudiantes, docentes y autoridades. Es importante realizar pruebas piloto y establecer mecanismos de monitoreo para detectar inconsistencias o fraudes, además de integrar tecnologías emergentes como la autenticación biométrica para mayor precisión. Se debe capacitar al personal y garantizar soporte técnico continuo para resolver incidencias, mientras se evalúa el impacto económico del sistema. El diseño debe permitir futuras ampliaciones y garantizar la sostenibilidad del sistema a largo plazo, manteniendo la transparencia y confiabilidad en la gestión de asistencias y permisos.

# Bibliografía

Aprendizaje, U. d. (2015). *repositorio uapa* . Obtenido de https://repositorio-uapa.cuaieed.unam.mx/repositorio/moodle/pluginfile.php/1453/mod\_resource/content/1/contenido/index.html#:~:text=En%20inform%C3%A1tica%2C%20una%20cola%20representa,nodos%20en%20el%20otro%20extremo.

Committee, S. E. (2005). *SWEBOK (Software Engineering Body of Knowledge).* USA: IEEE Computer Society.

Osvaldo Cairó & Silvia Guardati. (2006). *Estructura de Datos.* Mexico: McGrawHill.

Romero, F. V., & Salazar, P. J. (2024). *Implementación de un Sistema web para optimizar el funcionamiento de la gestión logística del área de almacén de la cadena de Restaurantes Vista al Mar de la ciudad de Lima - 2024.* Lima - Perú: Universidad Tecnoclogíca del Perú.