CAPÍTULO 1

# **1.1. Introducción**

Las áreas de tecnologías de información son claves para el fortalecimiento de nuevos cambios que apunten a esquemas de organización del trabajo de forma no solo eficientes sino también menos costosas en la utilización de tiempo y recursos, para ello se requiere el desarrollo de aplicaciones informáticas dirigidas a absorber tanto el conocimiento como los procesos, mediante la creación de soluciones integradas, orientadas a facilitar tareas además de consolidar gestión.

En la nueva era, la información y las comunicaciones, son factores extremadamente claves en los procesos de producción y obtención de mejoras. Las Tecnologías de la Informática y las Comunicaciones (TIC) han demostrado ser instrumentos que pueden contribuir al logro de amplios objetivos nacionales, tanto sociales como económicos, en la medida en que los estados las incorporen a las principales políticas y programas de desarrollo de sus naciones. El Colegio Nacional de EMD Prof. Atanasio Riera del Área 1 siempre se encuentra con el propósito realizar una mejora tecnológico para permitir la optimización de procesos de gestión del mismo.

A lo largo del proceso de desarrollo del software, basado en las necesidades del colegio, se realizaron sesiones de trabajo con los jefes de estudios para la toma de requerimientos. Í

Actualmente, se considera que el ambiente educativo puede contar con nuevas herramientas para facilitar la comunicación y el desarrollo de las diferentes actividades que se llevan a cabo hasta el momento; sin embargo, en el Colegio Nacional de EMD Prof. Atanasio Riera del Área 1, la implementación y utilización de dichas herramientas es todavía baja.

El actual auge de internet ha permitido minimizar la brecha digital sobre muchos procesos administrativos, permitiendo mejor organización, optimización, automatización y agilización de dichos procesos.

# **1.2 – JUSTIFICACION**

La tecnología avanza cada día más y el ser humano no puede quedarse atrás, por eso es necesario implementar sistemas que acompañen este avance y minimicen cada vez más las operaciones repetitivas del ser humano. En este contexto desarrollaremos un sistema informático que optimice las funciones de la Jefatura de Estudios en cuanto a documentaciones que hacen a su labor.

# El proyecto tiene la facultad de guardar todos los datos de los alumnos con su ausencia justificada e injustificada en una base de datos que guarda todas las informaciones de los mismos. Siendo administrada por los jefes de estudio.

# **1.3-PLANTEAMIENTO DEL PROBLEMA**

El Colegio del Área 1 no consta de un sistema que facilite a los jefes de estudio y a los docentes en general a administrar y registrar de una manera más ágil las ausencias de los Bachilleratos Técnico y Cientifico.

El jefe de estudio gestiona las ausencias, las faltas cometidas por los alumnos y todo lo registra en una planilla, además recibe los justificativos y documentos varios de los alumnos, pero al momento de buscar los registros esto tiene un tiempo de búsqueda considerable, y los documentos guardados tienen la posibilidad de perderse o dañarse.

Para paliar esta situación se plantea el desarrollo de un sistema informático que agilice el trabajo del jefe de estudio, reduzca el tiempo de búsqueda manual de los escritos, digitalice los documentos para evitar pérdidas y el uso de papeles.

**1.4–IMPACTO SOCIAL**

Tendrá un impacto en la agilidad y eficacia del trbajo de los profesores celadores donde ellos mismos podrá brindar asistencia al alumnado y Profesores

**1.5 OBJETIVOS**

**1.5.1 objetivo General**

Desarrollar e implementar un sistema informático que proporcione un control y gestión de Datos tanto personales como también Datos de los alumnos para favorecer el trabajo a los jefes de estudio.

**1.5.2 Objetivo Especifico**

* Conceptualizar y planificar un proyecto de software usando métodos, técnicas y herramientas adecuadas al contexto, para elaborar el plan de proyecto.
* Diseñar la base de datos usando métodos y herramientas, para generar un diseño adecuado del mismo.
* Diseñar los requisitos de especificación del sistema.
* Implementar y realizar las pruebas del sistema.

**1.6 Motivación**

Los tiempos cambian, todo avanza es nuestra motivación para realizar el proyecto, Ahora todo cambia debemos avanzar con el tiempo, entonces el propósito es ayudar a nuestros profes con el trabajo que ellos ejercen y adecuarlos a la tecnología de hoy.

**1.7 Organización del trabajo**

El trabajo los dividimos en 3 grupos:

Programadores: Elias Ojeda y Pedro Villagra

Capacitadores: Sandra Baez, Paloma Centurión y Lariza Sotto

Marco teorico: Eugenio Santa cruz y Rodrigo Coronel

**Capitulo II**

**2.1. Sistema**

Un sistema de información es un conjunto de elementos organizados y orientados al tratamiento y administración de datos e información para cubrir una necesidad u objetivo. Se caracteriza por la eficiencia en la que procesan los datos con relación a un área en específico.

Los elementos que componen un sistema de información son las personas, los datos, las actividades o técnicas de trabajo y los recursos materiales informáticos o de comunicación en general.

Todos estos elementos interactúan para procesar los datos (incluidos los procesos manuales y automáticos), y dan lugar a una información más elaborada que se distribuye de la mejor manera posible en una determinada organización y en función de sus objetivos.

Un sistema de información se destaca por su facilidad de uso, flexibilidad, mantenimiento automático de los registros, apoyo en toma de decisiones críticas y mantener el anonimato en informaciones no relevantes.

Un sistema de información debe cumplir con los siguientes componentes básicos interactuando entre sí:

* El **Hardware**, equipo físico utilizado para procesar y almacenar datos,
* El **Software** y los procedimientos utilizados para introducir, transformar y extraer información,
* Los **Datos** que representan las actividades de la institucion,
* Las **Personas** que desarrollan, mantienen y utilizan el sistema.

**2.2. Sistema Gestor de base de datos**

Un **sistema gestor de base de datos** (**SGBD**) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos. Los usuarios pueden acceder a la información usando herramientas específicas de consulta y de generación de informes, o bien mediante aplicaciones al efecto.

Estos sistemas también proporcionan métodos para mantener la integridad de los datos, para administrar el acceso de usuarios a los datos y para recuperar la información si el sistema se corrompe. Permiten presentar la información de la base de datos en variados formatos. La mayoría incluyen un generador de informes. También pueden incluir un módulo gráfico que permita presentar la información con gráficos y tablas.

Generalmente se accede a los datos mediante lenguajes de consulta, lenguajes de alto nivel que simplifican la tarea de construir las aplicaciones. También simplifican las consultas y la presentación de la información. Un SGBD permite controlar el acceso a los datos, asegurar su integridad, gestionar el acceso concurrente a ellos, recuperar los datos tras un fallo del sistema y hacer copias de seguridad. Las bases de datos y los sistemas para su gestión son esenciales para cualquier área de negocio, y deben ser gestionados con esmero.

Existen componentes del sistema de gestor de la base de datos, podemos encontrar en ellas:

* El motor de la base de datos: Acepta peticiones lógicas, los otros subsistemas del SGBD, las convierten en un equivalente físico y accede a la base de datos, y diccionario de datos en el dispositivo de almacenamiento.
* El subsitemas de definición de datos: Ayuda a crear y mantener el diccionario de datos y define la estructura del fichero, que soporta la base de datos.
* El subsitema de manipulación : Los datos que el usuario que puede añadir, cambiar y borrar información de la base de datos y la consulta para extraer información. El subsitema de manipulación de datos suele ser la interfaz principal del usuario de con la abse de datos. Permite al usuario especificar sus requisitos de la información desde un punto de vista lógico.
* El subsistema de generación de aplicaciones: Contiene utilidades para ayudar a los usuarios en el desarrollo de aplicaciones. Usualmente proporciona pantallas de entrada de datos, lenguajes de programación e interfaces.
* El subsistema de administración: ayuda a gestionar a la base de datos ofreciendo funcionalidades como almacenamiento y recuperación, gestión de la seguridad, optimización de preguntas, control de concurrencia y gestión de cambios.

**2.3 Base de datos**

Una **base de datos** es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En este sentido; una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta. Actualmente, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital, siendo este un componente electrónico, por tanto se ha desarrollado y se ofrece un amplio rango de soluciones al problema del almacenamiento de datos.

Hay programas denominados sistemas gestores de bases de datos, abreviado SGBD (del inglés *Database Management System* o DBMS), que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Las propiedades de estos DBMS, así como su utilización y administración, se estudian dentro del ámbito de la informática.

Las aplicaciones más usuales son para la gestión de empresas e instituciones públicas; También son ampliamente utilizadas en entornos científicos con el objeto de almacenar la información experimental.

**2.4 Sistemas informáticos**

Un **sistema informático** (SI) es un sistema que permite almacenar y procesar información; es el conjunto de partes interrelacionadas: *hardware*, *software* y personal informático. El *hardware* incluye computadoras o cualquier tipo de dispositivo electrónico, que consisten en procesadores, memoria, sistemas de almacenamiento externo, etc. El software incluye al sistema operativo, *firmware* y aplicaciones, siendo especialmente importante los sistemas de gestión de bases de datos. Por último, el soporte humano incluye al personal técnico que apoyan y mantienen el sistema (analistas, programadores, operarios, etc.) y a los usuarios que lo utilizan.

**2.4 Metodología de desarrollo de Software**

Una metodología de desarrollo de software se refiere a un *framework* (entorno o marco de trabajo) que es usado para estructurar, planear y controlar el proceso de desarrollo en sistemas de información.

A lo largo del tiempo, una gran cantidad de métodos han sido desarrollados diferenciándose por su fortaleza y debilidad.

El *framework* para metodología de desarrollo de *software* consiste en:

* Una **filosofía de desarrollo de programas de computación** con el enfoque del proceso de desarrollo de *software*
* Herramientas, modelos y métodos para asistir al proceso de desarrollo de *software*

Estos *frameworks* son a menudo vinculados a algún tipo de organización, que además desarrolla, apoya el uso y promueve la metodología.

**2.5 Metodologías agiles**

Se me suele preguntar la siguiente cuestión: **¿Qué es la metodología ágil?** Es simple, ágil es una palabra que la industria de la tecnología de la información usa para describir un método alternativo de gestión de proyectos.

El método Ágil es un proceso que permite al equipo dar respuestas rápidas e impredecibles a las valoraciones que reciben sobre su proyecto. Crea oportunidades de evaluar la dirección de un proyecto durante el ciclo de desarrollo. Los equipos evalúan el proyecto en reuniones regulares, llamadas sprints o iteraciones.

El método ágil es un **proceso de empoderamiento** que ayuda a las empresas a diseñar y crear el producto idóneo. El proceso de gestión es muy beneficioso para las compañías de software porque les permite analizar y mejorar su producto durante el desarrollo del mismo. Esto da a las empresas la capacidad de fabricar un producto valioso, de manera que se mantengan competitivas en el mercado.

**2.5.1 Origen De Las Metodologías Agiles De Desarrollo De Software**

De acuerdo a (Letelier & Penades, 2006) En un proceso de software existen numerosas propuestas metodológicas que inciden en distintas dimensiones del transcurso de desarrollo. Por una parte tenemos aquellas propuestas más tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, las herramientas y notaciones que se usarán. Estas propuestas han demostrado ser efectivas y necesarias en un gran número de proyectos, pero también han presentado problemas en otros. Una posible mejora es centrarse en otras dimensiones, como por ejemplo, el factor humano o el producto software. Esta es la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas.

**MANIFIESTO ÁGIL**

En una reunión celebrada en febrero de 2001 en Utah-EEUU, nace el término “Ágil” aplicado al desarrollo de software. En esta reunión participaron un grupo de 17 expertos de la industria del software, su objetivo fue proyectar los valores y principios que deberían permitir a los equipos desarrollar software rápidamente y respondiendo a los cambios que puedan surgir a lo largo del proyecto. Tras esta reunión se creó The Agile Alliance, una organización, sin ánimo de lucro dedicado a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones para que adopten dichos conceptos. El punto de partida fue el Manifiesto ágil, un documento que resume la filosofía ágil.

Este manifiesto ágil comienza enumerando los principales valores del desarrollo ágil, como son:

* Al Individuo y las iteraciones del equipo de desarrollo sobre el proceso y las herramientas.
* Desarrollar software que funciona más que conseguir una buena documentación.
* La colaboración con el cliente más que la negociación de un contrato.
* Responder a los cambios más que seguir estrictamente un plan.

Los valores anteriores inspiran los doces principios del manifiesto ágil, son características que diferencian un proceso ágil de uno tradicional. Los Dos primeros son generales y resumen gran parte del espíritu ágil, Estos son:

* La Prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
* Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
* Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
* La Gente del negocio y los desarrolladores deben trabajar juntos a lo largo e proyecto.
* Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo
* El Diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
* El software que funciona es la medida principal de progreso.
* Los Procesos Ágiles promueven un desarrollo sostenible. Los promotores,desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
* La Atención continua a la calidad técnica y al buen diseño mejora la agilidad.
* La Simplicidad es esencial.
* Las Mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.
* En Intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivos y según esto ajusta a su comportamiento.

Aunque los creadores e impulsadores de las metodologías agiles más populares han suscrito el manifiesto ágil y coinciden con los principios enunciados anteriormente, cada metodología tiene c.racterísticas propias.

Tabla 2.1.Comparación entre metodologías ágiles y tradicionales

|  |  |
| --- | --- |
| Metodologías Ágiles | Metodologías Tradicionales |
| Basadas en heurísticas provenientes de prácticas de producción de código. | Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo. |
| Preparados para el cambio durante el proyecto. | Cierta resistencia a los cambios. |
| Reglas de trabajo impuestas internamente (Por equipos). | Reglas de trabajo impuestas externamente. |
| Proceso menos controlado, con pocos principios. | Procesos mucho más controlados con numerosas políticas/ Normas. |
| El Cliente es parte del desarrollo. | El Cliente interactúa con el equipo de desarrollo mediante reuniones en determinadas etapas del proceso.. |
| Pocos Artefactos | Más Artefactos |
| Pocos Roles | Más Roles |

En la tabla 2.1 se observa la comparación entre las metodologías ágiles y las metodologías tradicionales, mediante esta se afirma que las metodologías ágiles son más orientadas a proceso de desarrollo de software con pocas semanas de desarrollo y bajos niveles de formalización en la documentación requerida.

**2.5.2 Principales Metodologías Ágiles**

Las Metodologías Ágiles resuelven los problemas surgidos, posteriormente, a la masificación del uso del computador personal, dado que las expectativas y necesidades por parte de los usuarios se hicieron más urgentes y frecuentes.

Fue así como al comienzo de los 90 surgieron propuestas metodológicas para lograr resultados más rápidos en el desarrollo del software sin disminuir su calidad. (Orjuela Duarte & Rojas C, 2008)

Entre las principales metodologías ágiles se encuentran las siguientes:

* PROGRAMACIÓN EXTREMA (XP): La Programación Extrema o Extreme Programing, es un enfoque de la ingeniería de software formulado por Kent Beck, se considera el más destacado de los procesos ágiles de desarrollo de software. Al igual que estos, la programación extrema se diferencia de los métodos tradicionales principalmente en que presenta más énfasis en la adaptabilidad que en la previsibilidad.
* SCRUM: Según (Orjuela Duarte & Rojas C, 2008), está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos: El Desarrollo del software se realiza mediante iteraciones denominadas Sprint, con una duración de 30 días, el resultado de cada Sprint es un incremento ejecutable que se muestra al cliente. La Segunda Característica importante son las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración.
* CRYSTAL METODOLOGÍAS: Se Trata de un conjunto de metodologías para el desarrollo de software caracterizadas por encontrarse centradas en las personas que componen equipo y la reducción al máximo del número de artefactos producidos. El desarrollo de software se considera un juego cooperativo de invención y Comunicación, limitado por los recursos a utilizar. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo definidas. Estas políticas dependerán del tamaño del equipo, estableciéndose una clasificación por colores por ejemplo: Crystal Clear (3 a 8 integrantes), Crystal Orange (25 a 50 integrantes). (Orjuela Duarte & Rojas C, 2008)
* ADAPTIVE SOFWARE DEVELOPMENT (ASD): Presupone que las necesidades del cliente son cambiantes. La Iniciación de un proyecto involucra definir una misión para él, determinar las características, las fechas y descomponer el proyecto en una serie de pasos individuales, cada uno de los cuales puede abarcar entre cuatro y ocho semanas. Los pasos iniciales deben verificar el alcance del proyecto, los tardíos tienen que ver con el diseño de la arquitectura, la construcción del código, la ejecución de las pruebas finales y el despliegue. (Amaro Calderón & Valverde Rebaza, 2007)
* PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE: En (Torossi, 2012), se dice que el Proceso Unificado, es un “Conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema software”. Este proceso se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo consta de cuatro fases como son: Inicio, Elaboración, Construcción y Transición.
* DYNAMIC SYSTEMS DEVELOPMENT METHOD: Es la única de las metodologías planteadas surgida de un consorcio formado originalmente por 17 miembros fundadores en enero de 1994. El Objetivo del consorcio era producir una metodología de dominio público que fuera independiente de las herramientas y que pudiera ser utilizada en proyectos de tipo RAD (Rapid Application Development). En DSDM se define cinco fases en la construcción de un sistema, las mismas son: Estudio de factibilidad, Estudio del Negocio, Iteración del modelo funcional, Iteración del diseño y Construcción, Implantación. La Adecuación de DSDM para desarrollo rápido está suficientemente aprobada y se ha aplicado a proyectos grandes y pequeños. (Amaro Calderón & Valverde Rebaza, 2007)

**2.5.3 Metodología Programación Extrema (Xp)**

**ORIGEN PROGRAMACION EXTREMA (XP)**

Nace de la mano de Kent Beck en el verano de 1996, cuando trabajaba para Chrysler Corporation. Él tenía varias ideas de metodologías para la realización de programas que eran cruciales para el buen desarrollo de cualquier sistema. Las ideas primordiales de sus sistemas las comunico en las revistas C++ Magazine en una entrevista que esta le hizo el año 1999.

**¿QUE ES PROGRAMACION EXTREMA O XP?**

Es una Metodología ligera de desarrollo de aplicaciones que se basa en la simplicidad, la comunicación y la realimentación del código desarrollado.

**OBEJTIVOS DE XP**

* La Satisfacción del cliente.
* Potenciar el trabajo en grupo.
* Minimizar el riesgo actuando sobre las variables del proyecto: costo, tiempo, calidad, alcance.

**CARACTERÍSTICAS**

* Metodología basada en prueba y error para obtener un software que funcione realmente.
* Fundamentada en principios.
* Está orientada hacia quien produce y usa software (el cliente participa muy activamente).
* Reduce el coste del cambio en todas las etapas del ciclo de vida del sistema.
* Combina las que han demostrado ser las mejores prácticas para desarrollar software, y las lleva al extremo.
* Cliente bien definido.
* Los requisitos pueden cambiar.
* Grupo pequeño y muy integrado (2-12 personas).
* Equipo con formación elevada y capacidad de aprender

Capítulo 3

Descripción del sistema

3.1. Introducción.

Este capítulo tiene como objetivo la descripción detallada del sistema, exponiendo su composición, funcionalidades, modos de configuración y comportamiento adecuado a cada posible escenario. Posteriormente se describe las funcionalidades de la aplicación encargada de gestionar el sistema. La metodología aplicada para el desarrollo del sistema fue la programación extrema (extreme Programming). Es el más destacado de los procesos \_agiles de desarrollo de software. La programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos (Extreme Programing, 2013).

3.2. Proceso de desarrollo del sistema

En (Echeverry Tobón & Delgado Carmona, 2007)La Metodología Extreme Programmming o XP, está orientada al desarrollo de software cuando los requerimientos son ambiguos o rápidamente cambiantes asumiéndolos como algo natural, por lo que los programadores deben responder a estos cambios cuando el cliente lo solicite.

XP es para pequeños y medianos equipos basándose en la comunicación continúa entre todos los participantes, la simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Esta Metodología recomienda a seguir las siguientes prácticas, las cuales fueron implementadas para el desarrollo del sistema:

**3.2.1. Planificación:**

Es la Fase inicial de la metodología XP, donde se establece una comunicación continua entre el equipo de desarrollo y el cliente en este caso con los jefes de estudios del colegio Área 1, para obtener principalmente los requisitos del sistema. El sistema consta con las siguientes especificaciones

Reglas de negocio

* RN001: Función que genera un código secuencial único que idéntica al registro dentro de la tabla de la base de datos.
* RN002: Función que muestra lista de registros existentes especıﬁcos.
* RN003: Operación que muestra los datos de un registro especiﬁco existente dentro de los campos del formulario.
* RN004: Función que valida que el dato ingresado coincida con el dato solicitado en el formulario. El sistema no debe permitir el ingreso de datos distintos al dato requerido.
* RN005: Función que valida que el campo no este vacío.
* RN006: Función deﬁne privilegios de acceso a cada usuario.
* RN007: Operación que vacía los campos de texto.
* RN008: Operación de almacenar datos de un formulario en la base de datos al seleccionar la función en el botón correspondiente.

Requisitos funcionales

* RF001: Operación que permite ingresar al sistema, bajo validación de usuario y contraseña al dar clic en botón correspondiente.
* RF002: Operación que re direccionar a sección de registro de usuarios invitados.
* RF003: Función que permite registrar usuarios al sistema.
* RF004: Operador que registra usuario en la base de datos luego de completar el formulario y seleccionar la función en el botón correspondiente.
* RF005: Función que permite mostrar variables medidas por nodo especiﬁco al seleccionar la función en el sector de una imagen que este registrado.
* RF006: Indicador de nodo conectado/desconectado según su color(negro desconectado, verde conectado).
* RF007: Operación que permita realizar ﬁltros en los datos guardados en la base de datos.
* RF008: Función que muestra sección de la aplicación al dar clic en la opción correspondiente del menú´.
* RF009: Función que muestra gráﬁcos estadísticos de variables medidas.
* RF010: Función que muestra historias de mediciones de todo el sistema.
* RF011: Función que permite conﬁguracion de parámetros para cada nodo.
* RF012: Operación que permite activar/desactivar actuadores manualmente al dar clic en botón correspondiente.
* RF013: Función que automatiza la operación de un nodo al chequear el casilla correspondiente.
* RF014: Operador que permita cerrar sesión en la aplicación.
* RF015: Función que permite recorrer gráﬁcos estadísticos y exportar el mismo.
* RF016: Función que obtiene variables medidas por el nodo.
* RF017: Función que permite ﬁltrar datos almacenados en la base de datos.
* RF018: Función que permite cuales actuadores se activaran frente a un evento estando el nodo conﬁgurado en modo automático.
* RF019: Función que permite establecer acceso activar o inactivo a usuario del sistema.

**3.2.2. Diseño**

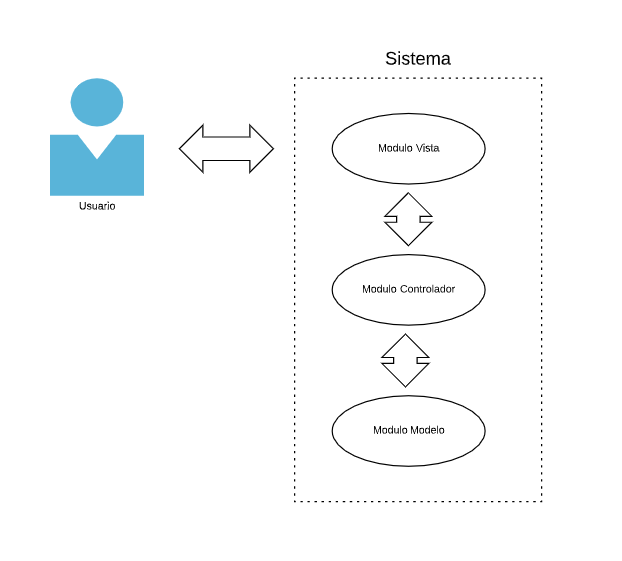
La Metodología XP hace especial énfasis en los diseños simples y claros. A continuación se observan los diagramas de casos de usos, que deﬁnen la naturaleza de cada caso de uso, son descripciones de pasos o actividades que deben realizarse por un actor determinado.

Diagrama de casos y usos.

A continuacion se observan los diagramas de casos de usos, que definen

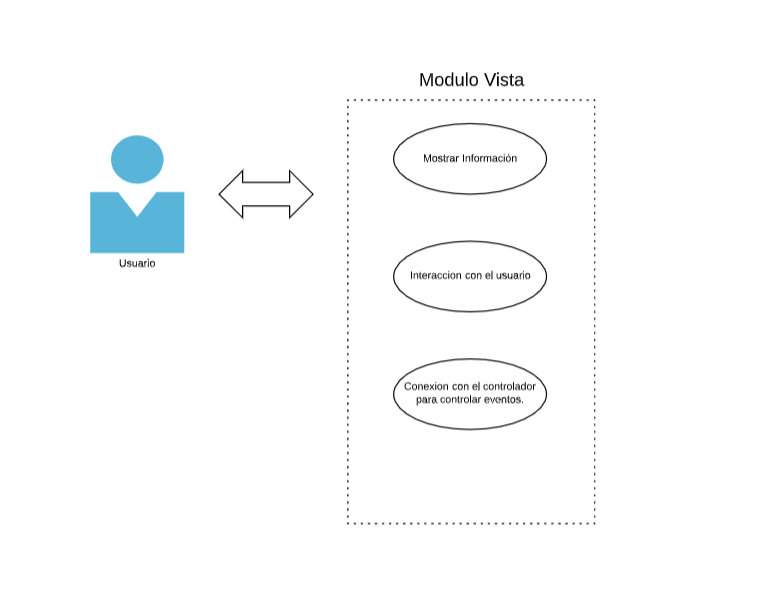
la naturaleza de cada caso de uso, son descripciones de pasos o actividades

que deben realizarse por un actor determinado.



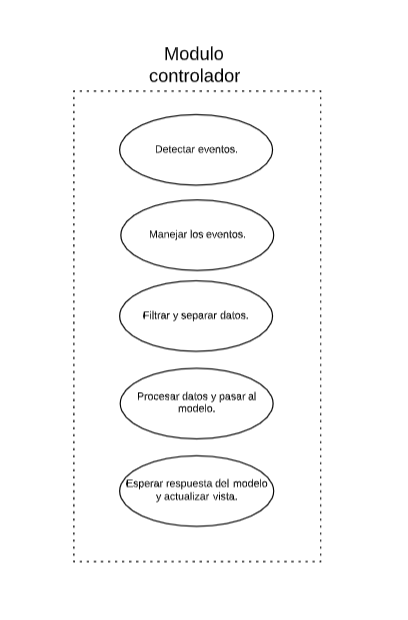
Modulo Vista

Este modulo provee al usuario una interfaz agradable, interactiva que le permite mostrar información, y manejar los datos respectivamente.



Modulo Controlador

Este modulo no es visible al usuario, pero se encarga del control de los eventos que realiza al usuario y procesar los datos atraves del modelo y luego devolver la respuesta adecuada para al usuario.



Modulo Modelo

Este no es visible al usuario, es la parte mas profunda del sistema, Este se encarga de procesar los datos que es pasado desde el controlador, de acuerdo a estos datos se preparan las consultas y se utilizan los DTO’s para devolver los datos en forma de objetos ya procesados.

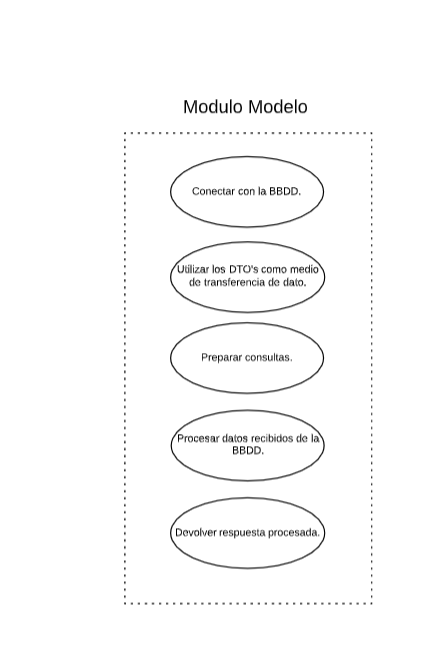
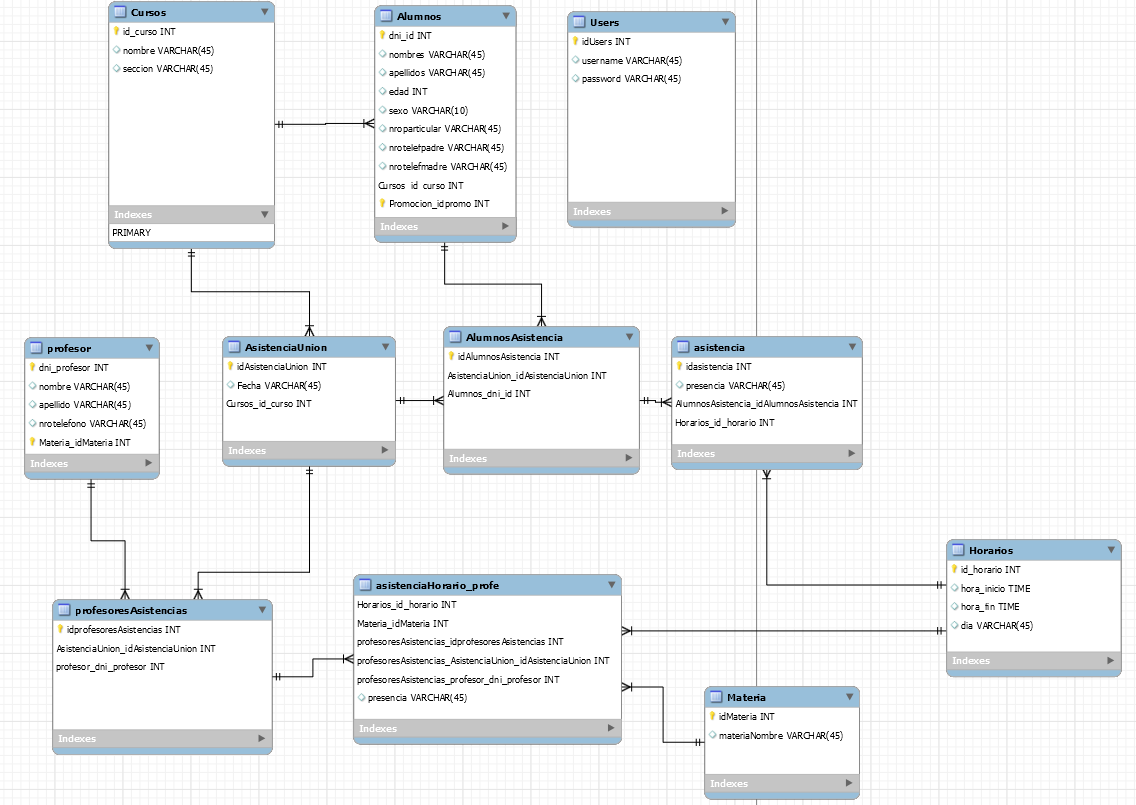


Diagram E/R del sistema



Este diagrama es la representación de como los datos son relacionados dentro del sistema, e indica como los datos son procesados para su guardado. Y también el proceso que requiere para unir y mostrar como datos útiles para el usuario.

3.2.3 Codificación

El diseño del sistema de gestión y control de asistencias de alumnos, fue realizado utilizando el lenguaje de programación JAVA en su versión 8, La utilización de la librería grafica JavaFX 2, y SQLite como base de datos relacional embebida con soporte para SQL.

El diseño contemplo 4 etapas importantes:

* Diseño de la base de datos.
* Diseño de un prototipo del sistema.
* Codificación del mismo.
* Pruebas.

3.2.4 Tecnologías utilizadas

JAVA es un lenguaje de programación orientado a objetos, En java todo se considera objetos, exceptuando los tipos primitivos. Este lenguaje nació con cinco objetivos principales:

* Debería usar el paradigma orientado a objetos.
* Debería permitir la ejecución de un mismo programa en diferentes sistemas operativos.
* Debería incluir por defecto un soporte para trabajo en red.
* Debería diseñarse para ejecutar código en remoto de forma segura.
* Debería ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos como c++.

Una de las características mas importantes en java es su independencia de plataforma este sigue la frase “Write once, run anuwhere”, El código de java solo debe ser escrito una vez para poder ser ejecutado en cualquier plataforma, Otras de sus características más importantes es su Recolector de basura, el cual se encarga de identificar los objetos que ya no se utilizan y eliminarlos de la memoria, Esto permite que el programador no tenga que preocuparse por la memoria, Java lo hará solo.

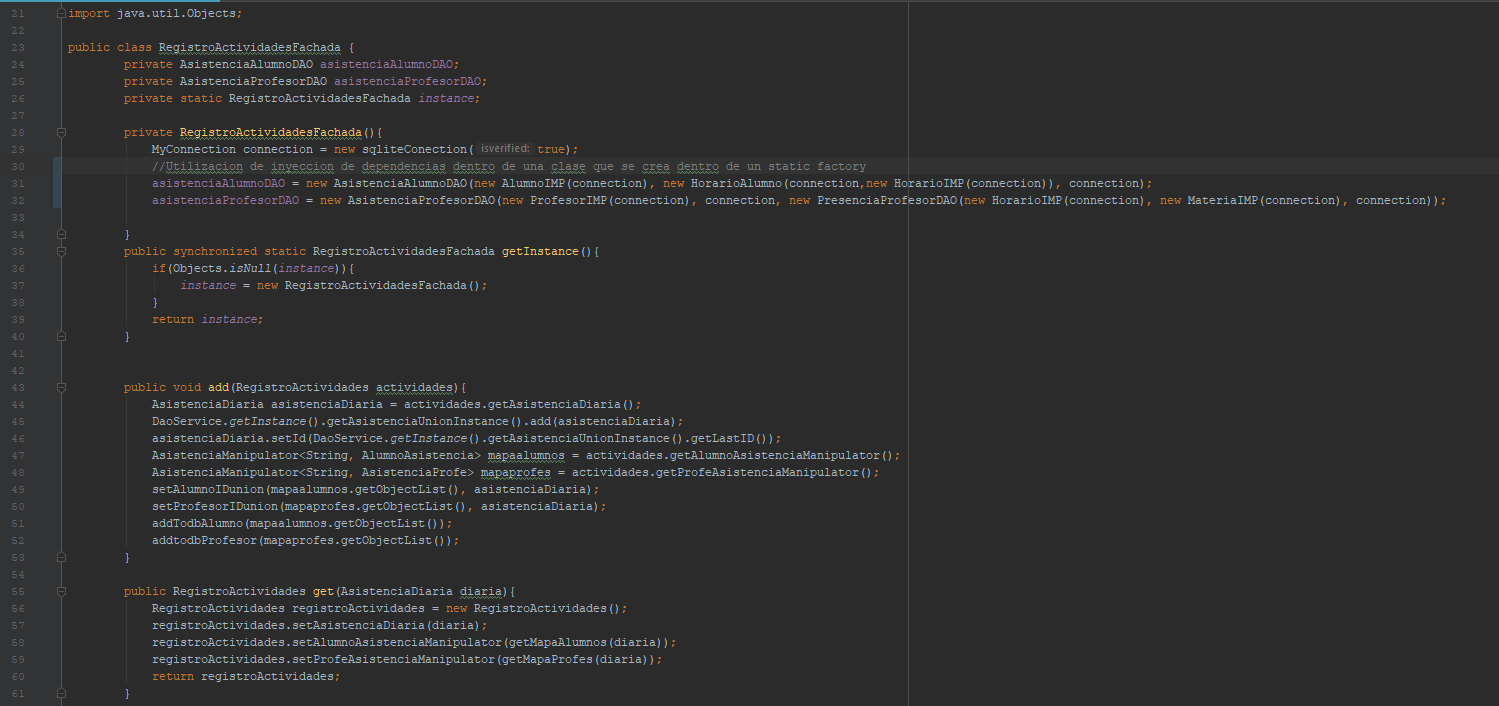
JavaFx forma parte del lenguaje, Este es una librería grafica que posee todo lo necesario para poder crear UI’s interactivas y atractivas, Ademas de usar como motor de dibujado OpenGL, esto permite aprovechar la potencia de la GPU del usuario.

SQLite es un motor de base de datos embebido que se distribuye como plugin del lenguaje a utilizar, Este motor crea la base de datos en la computadora del usuario, y utiliza el lenguaje SQL como lenguaje de consultas.

Se utilizaron patrones de diseños. Los cuales se describen a continuación:

* MVC: es un patrón de arquitectura de software, que separa los datos y la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el **modelo**, la **vista** y el **controlador**, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario. ​ Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y posterior mantenimiento.
* DAO: En software de computadores, un **objeto de acceso a datos** (en inglés, *data access object*, abreviado **DAO**) es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo.

Ahora se muestra una pequeña muestra del código y con su explicación de funcionamiento.



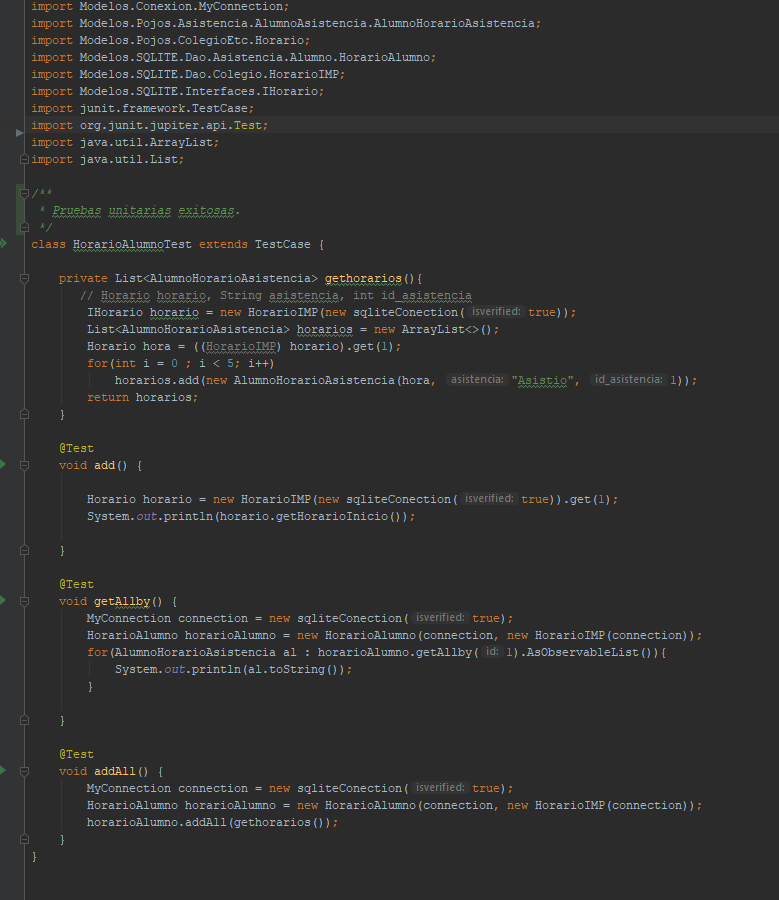
Este código de la clase RegistroActividadesFachada posee un constructor privado el cual se llama internamente desde un método sincronizado y estático, Esto permite únicamente llamar al nombre de la clase para la generación de dicho objeto. Este utiliza un “singleton” para que sea un único objeto durante todo el funcionamiento.

En el método add se pasa un objeto de RegistroActividades para ser añadido a la base de datos. Estos contienen tabla de datos, que son procesadas para añadir a la base de datos. Estos se procesan dividiéndolos en objetos más sencillos.

En el método get se pasa una AsistenciaDiaria el cual es un DTO asi como el otro, para poder obtener el id de la relación y poder obtener los datos relacionados de la base de datos y procesarlos retornarlo como salida, Todos estos procesos son ejecutados concurrentemente para evitar congelamientos en la ventana de usuario.

3.2.5. Pruebas

En esta fase se realizaron todas las pruebas unitarias a cada módulo/clases del sistema, Esto para prevenir problemas o funcionamiento no deseado durante el funcionamiento y fase de integración. Se utilizó JUnit como librería de debug para las pruebas unitarias, Todas las pruebas unitarias salieron exitosas en la primera prueba, Haciendo posible una integración exitosa en el sistema entero. Aquí un ejemplo de cómo es una prueba unitaria de una clase del sistema.



Las pruebas fueron realizadas método a método, para poder exponer posiblesg fallos.

Capitulo 4

El sistema se constituye con una sola parte principal el Software, el mismo esta compuesto por una aplicación para la gestión del sistema. La aplicación contiene funcionalidades de corroboran para la gestión del sistema, en ella se encuentran secciones particulares donde se puede realizar el registro de usuarios, configuración de cada nodo, consultas de historiales, monitoreo instantáneo de cada nodo activo, control de actuadores y servicio de notificaciones de eventos.

4.5.1. Módulo de consultas

Este módulo provee al usuario informaciones correspondientes de datos generados por cada nodo, como también conﬁguraciones almacenadas a respecto con parámetros, registros de usuarios, gráﬁcos estadísticos y nodos registrados.

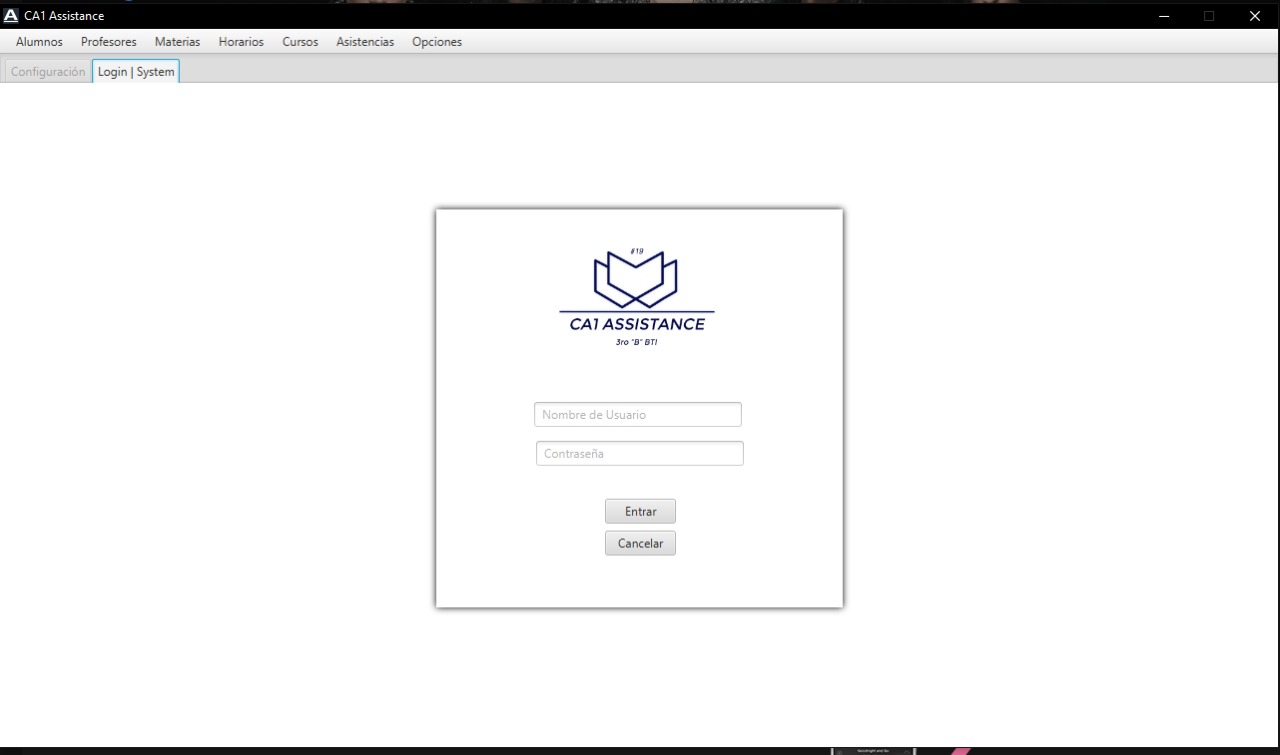
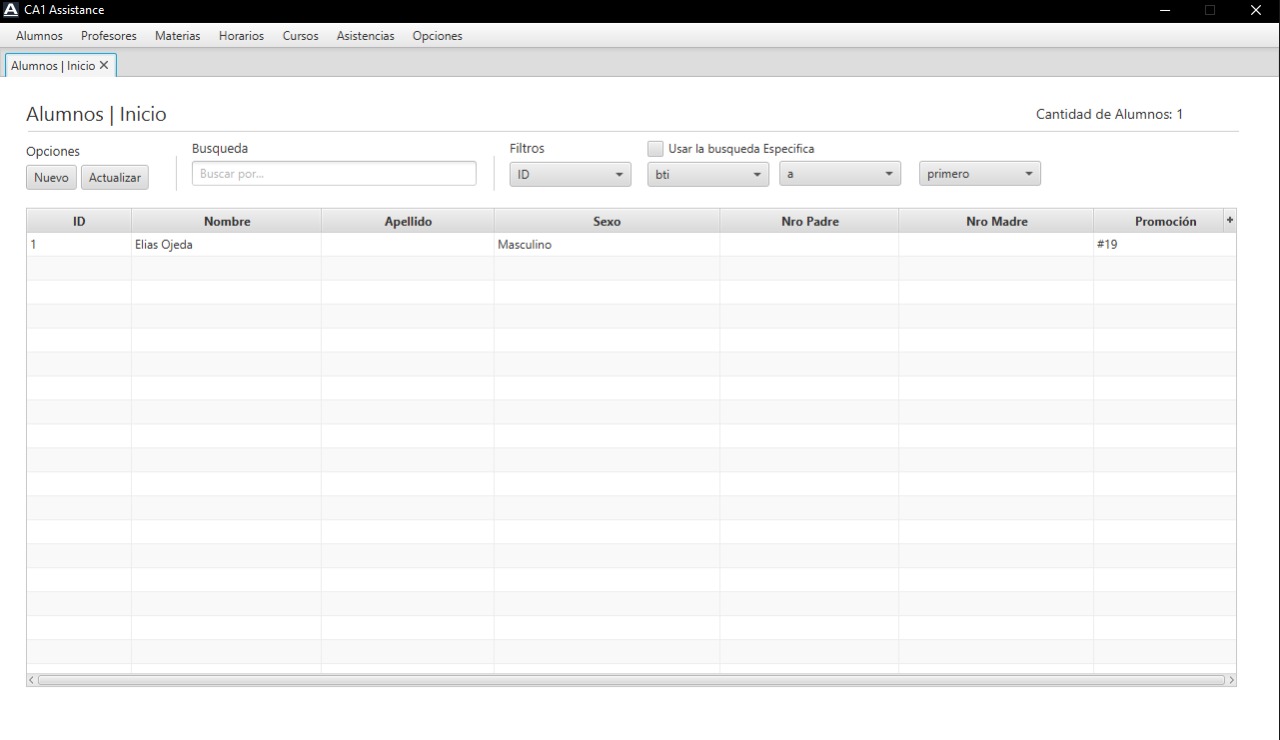


Figura 4.1: Login del sistema

El login del sistema sirve para evitar el acceso no autorizado a personas terceras al usuario, Además de la seguridad brinda privacidad única en cada instancia.



Figurar 4.2: Pantalla inicio de Alumnos.

La pantalla del listado de alumnos donde uno saber quien esta en el sistema, puede buscar utilizando las opciones de filtrado, agregar si es que no esta.

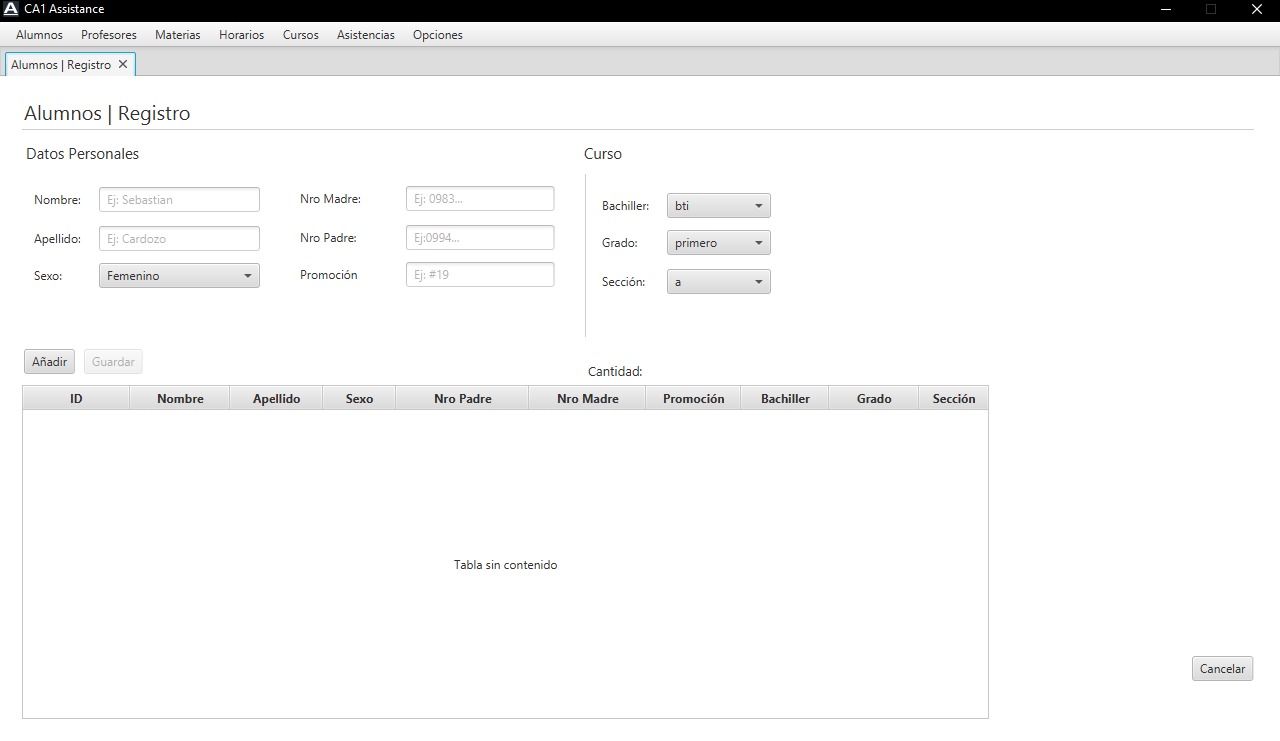


Figura 4.3: Pantalla registro de alumnos.

La pantalla registro de alumnos, permite agregar alumnos, cada alumno posee un id incremental único. Permite la selección de los bachilleres disponibles u o agregados.

Esto permite relacionar al alumno con su curso, y facilitar la obtención y manipulación de datos.

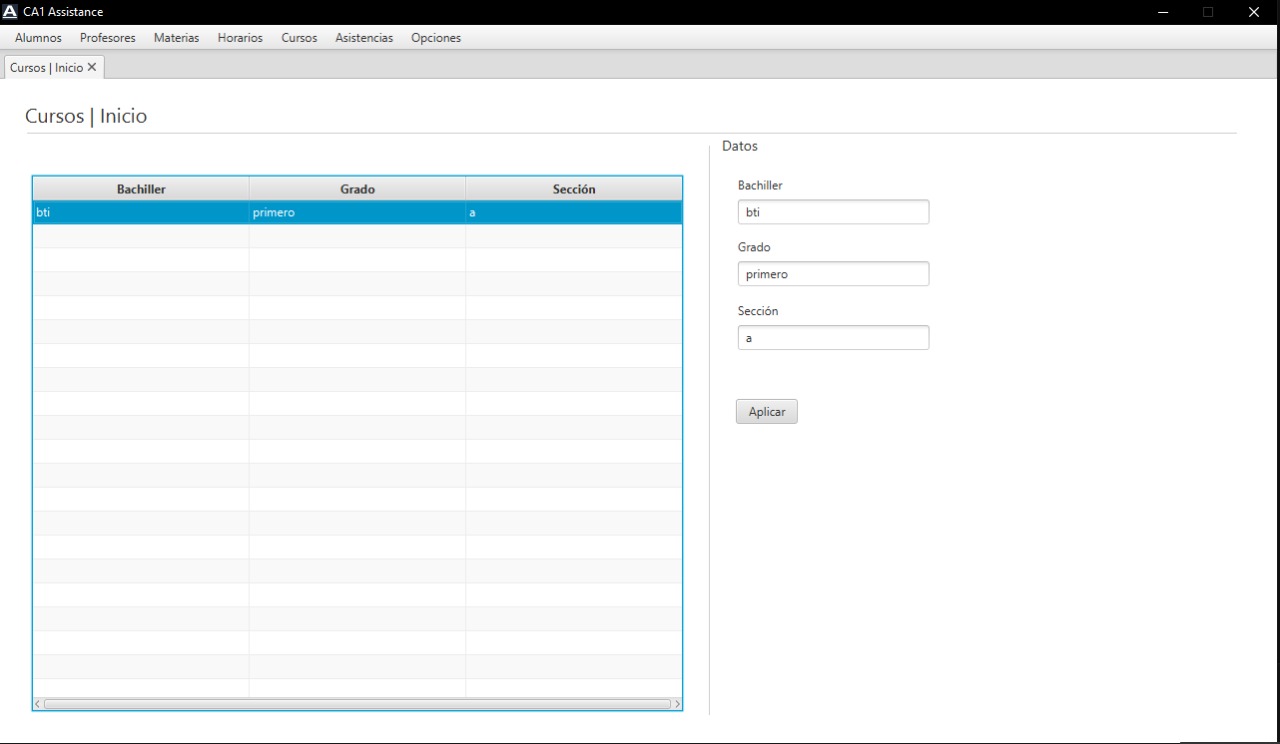


Figura 4.4: Pantalla Cursos Inicio

Esta pantalla permite agregar cursos de manera personalizada a las necesidades del Usuario, Al agregar cada curso este se verifica si es que ya existe en la base de datos, si no existe lo agrega, en caso contrario lo informa. También permite la edición de los datos con solo seleccionarlo.

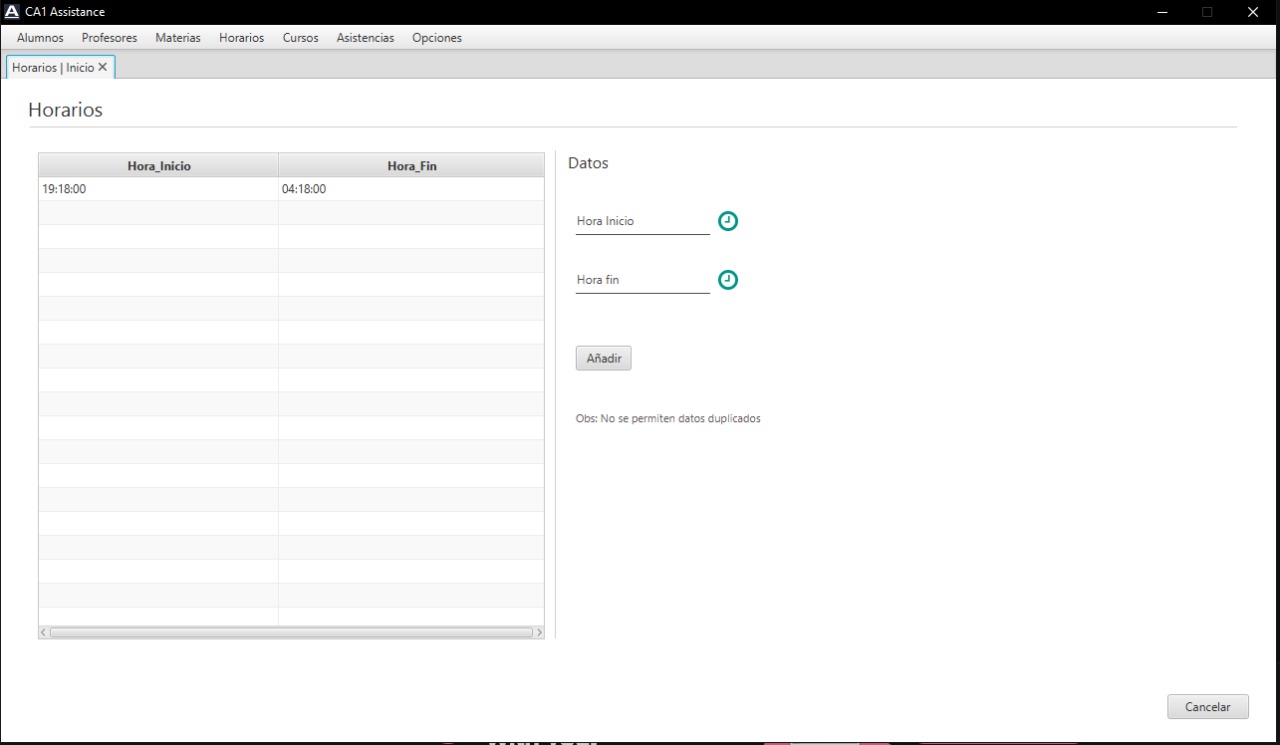


Figura 4.5: Pantalla de horarios.

Esta pantalla permite agregar horarios utilizando un selector de horario de acuerdo a la especificada en el sistema. También permite la edición de horarios.

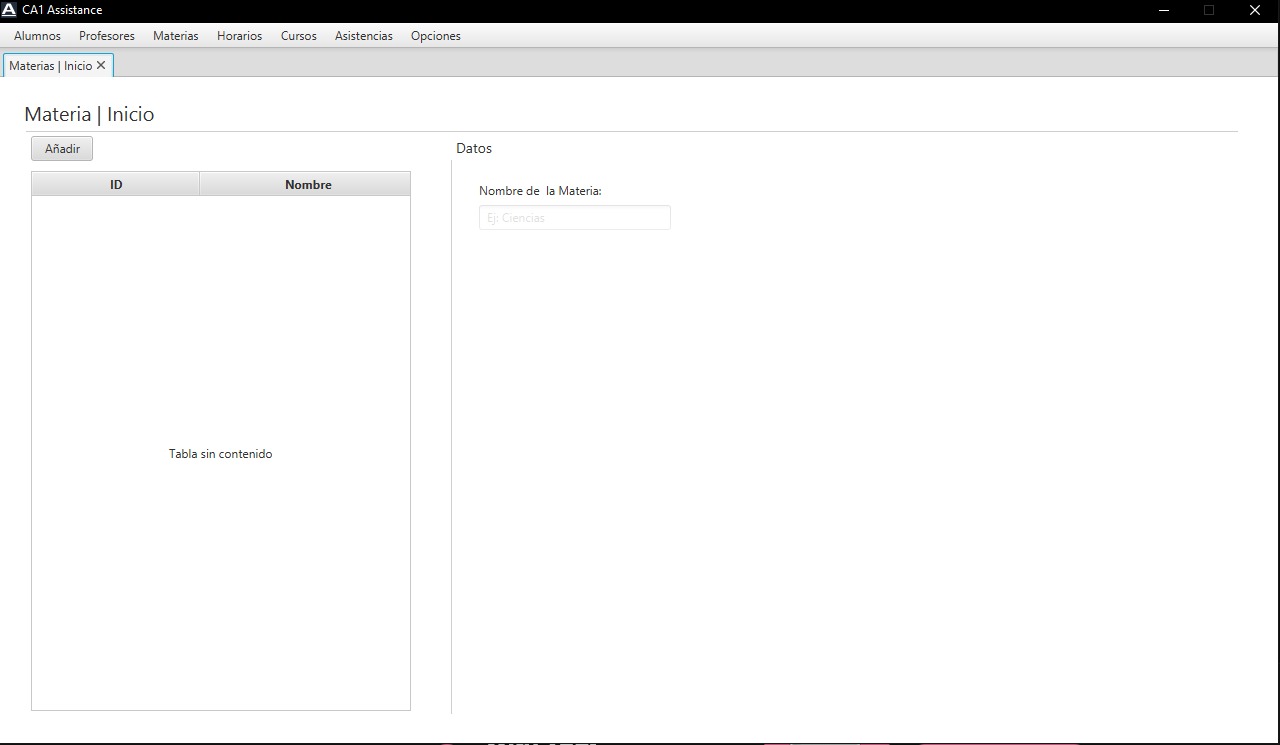


Figura 4.6: Pantalla de Materias

Esta pantalla permite agregar Materias utilizando el nombre de la materia, un id incremental y verifica si ya existe en la base de datos.

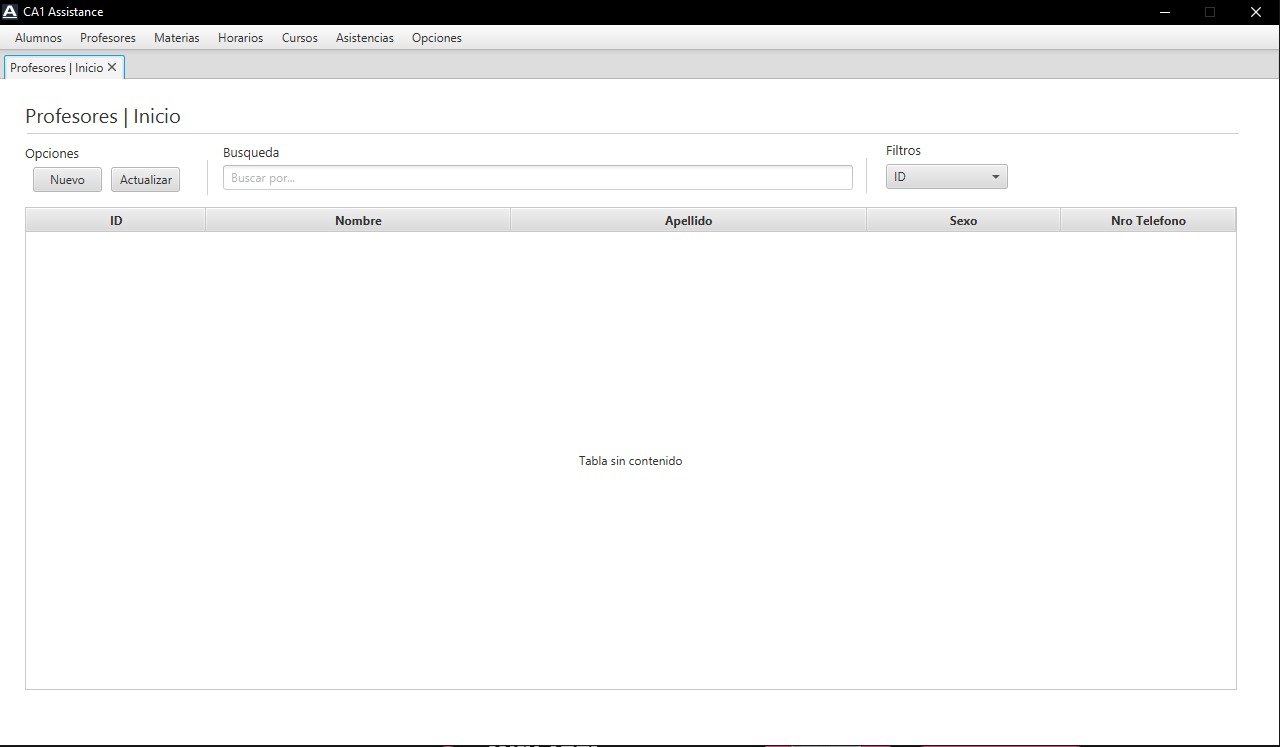
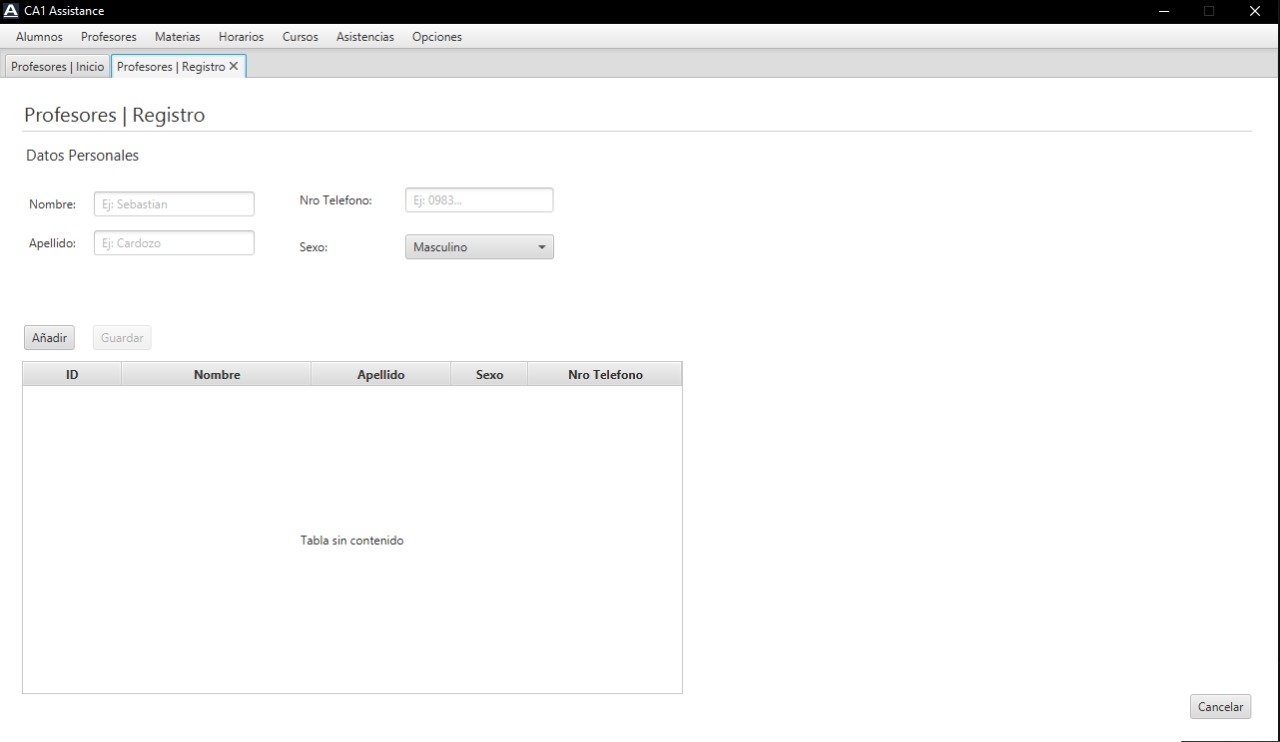


Figura 4.7: Pantalla de Profesores

Permite la búsqueda de información de profesores por medio del nombre o por otros filtros de búsqueda especificados en la imagen.



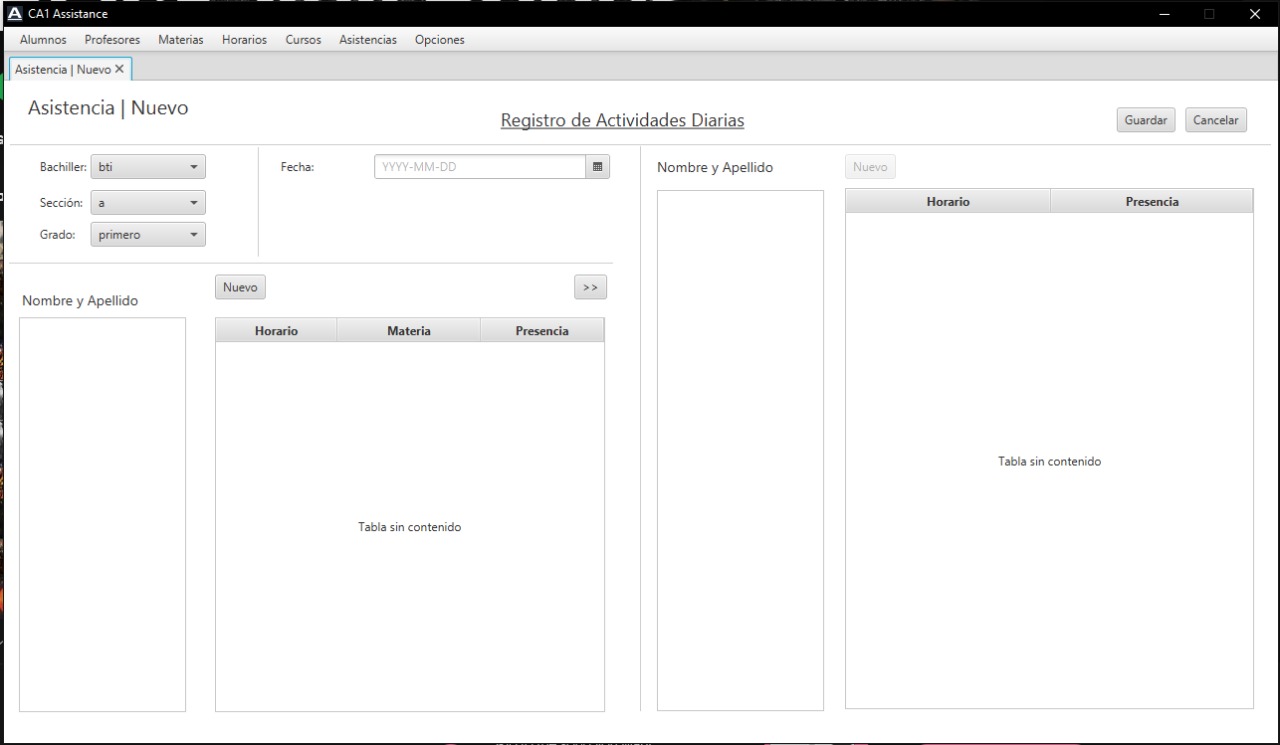
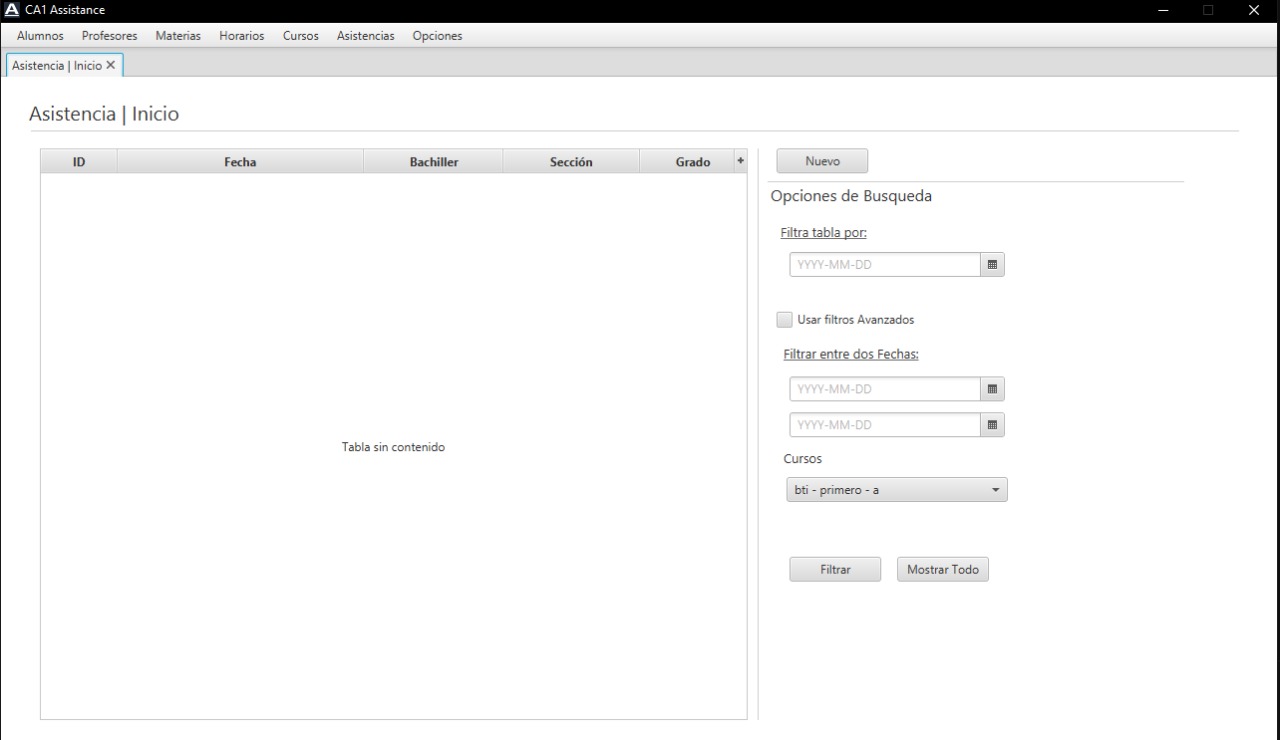


Figura 4.8: Pantalla de Asistencias

Esta pantalla permite agregar la asistencia de los alumnos como profesores, esta pantalla es una de las más importantes en el sistema.



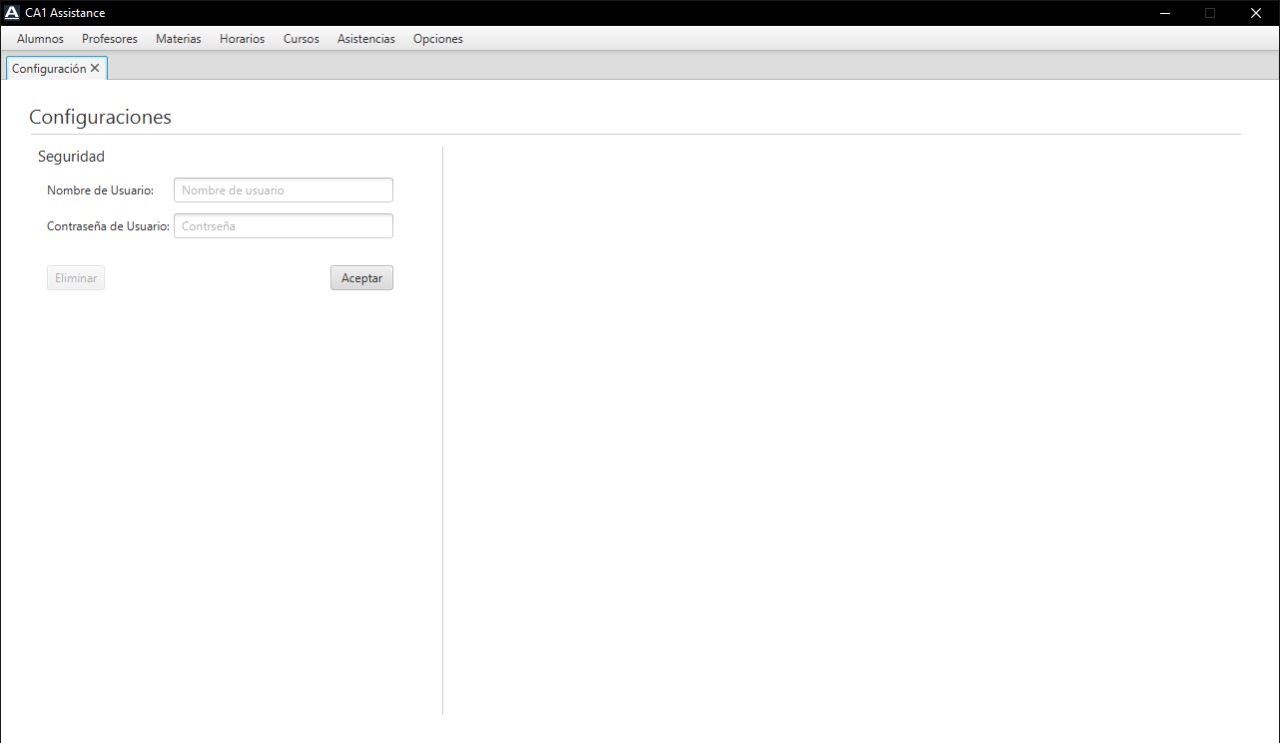
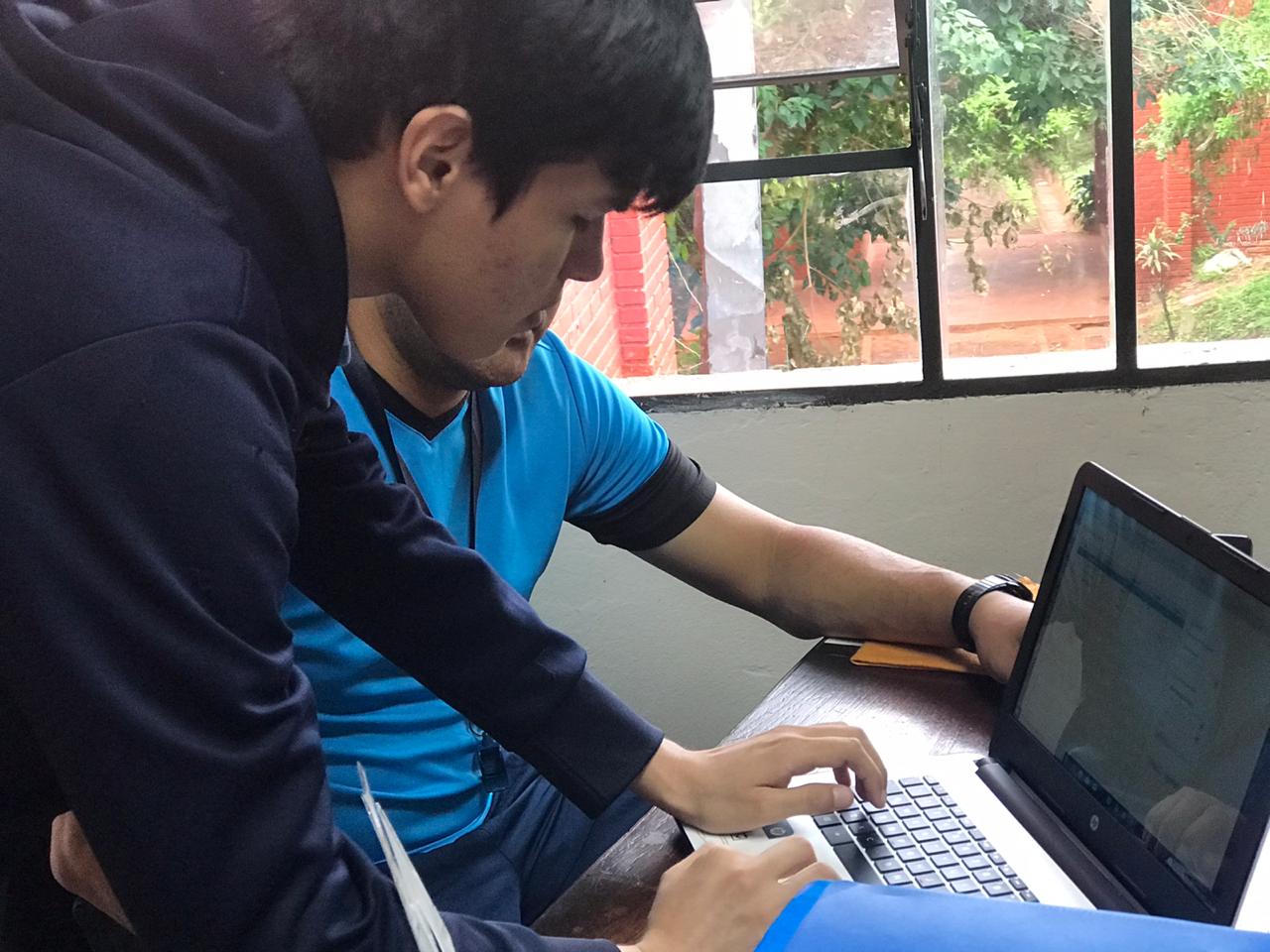
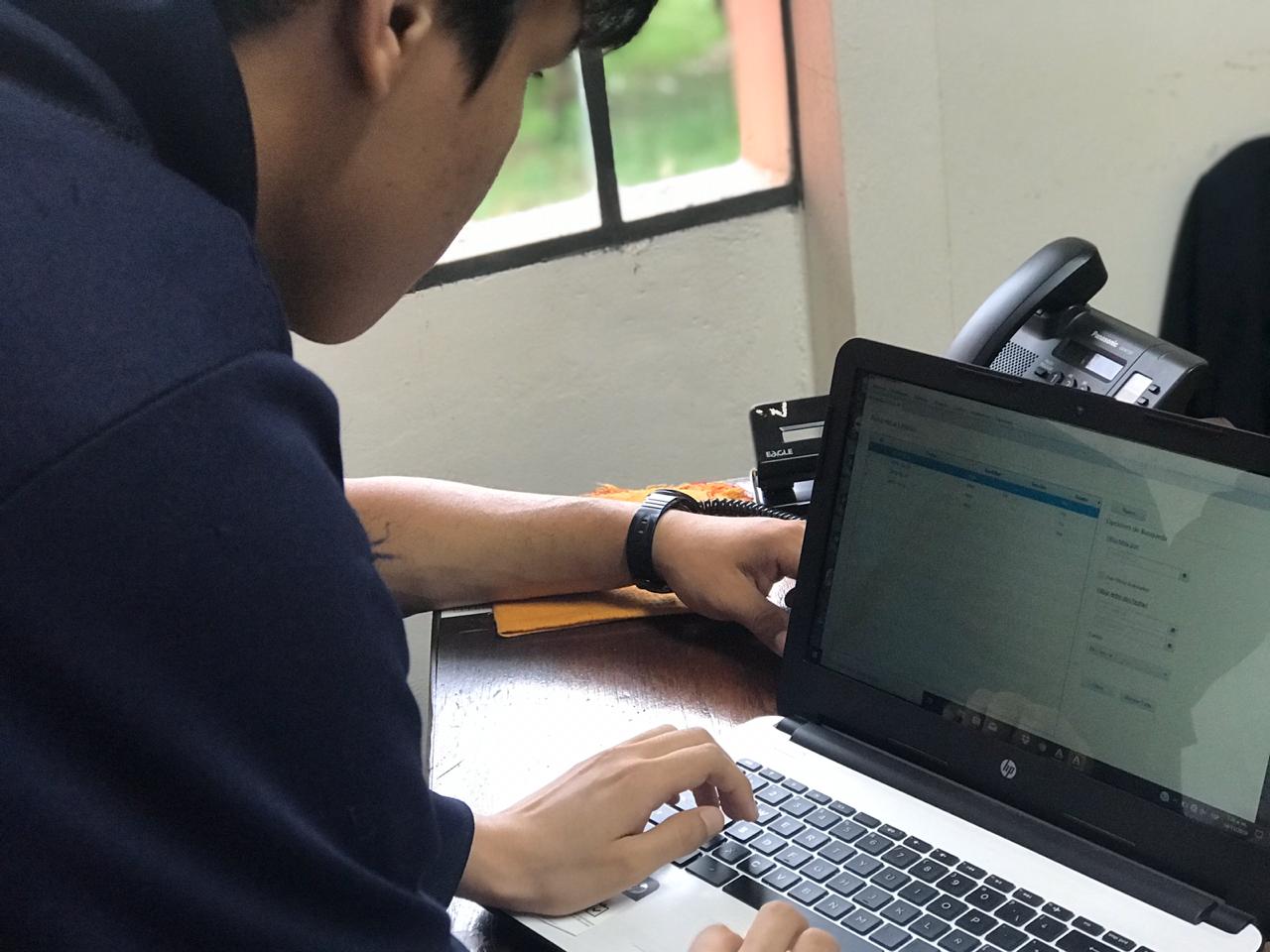


Figura 4.10: Pantalla de Configuración

**Capacitación al Usuario**

Esta capacitación es importante es aquí en donde el profe hace





**Encuesta al Profesor**

1. El sistema contiene todo lo que usted necesita para ejecutar su trabajo

Excelente

muy bueno

bueno

regular

malo

1. El sistema es de fácil acceso

Excelente

muy bueno

bueno

regular

malo

1. El sistema tiene un ambiente amigable

Excelente

muy bueno

bueno

regular

malo

1. Puede usarlo sin instrucciones escritas

Excelente

muy bueno

bueno

regular

malo

1. Has aprendido a utilizarlo rápidamente

Excelente

muy bueno

bueno

regular

malo

7. Estás satisfecho con el sistema

Excelente

muy bueno

bueno

regular

malo

8. Le ahorra tiempo en su trabajo

Excelente

muy bueno

bueno

regular

malo

